THE UNIVERSITY OF ALBERTA

THE TEACHER-AUTHORED INSTRUCTION MANAGER (TAIM): A COMPUTER MANAGED INSTRUCTION SYSTEM

by

© MARVIN LAWTON WESTROM

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

THE DEPARTMENT OF SECONDARY EDUCATION

EDMONTON, ALBERTA

FALL, 1973

ABSTRACT

The Teacher-Authored Instruction Managment System (TAIM) is a prototype instructional process. It utilizes computer programs which allow a teacher to input curriculum content (Displays) and Tests, and rules (Logic) for assigning these to individual students. The programs accept student responses, score tests, and keep student records. On the basis of these records and the Logic, individual computer printouts are assembled from the Displays and Tests for each registered student. Facility is provided to retrieve student scores and records of the lessons given to students, and for the teacher to override the specified Logic for individuals or groups.

This report presents a rationale, description and feasibility assessment of the prototype. The rationale is derived from principles of system theory and analyzes the instructional process in terms of information flow and storage. The description of the instructional process centers on the usage of the computer programs, but the architecture of these is discussed more fully by Zarsky (1973). The feasibility assessment is based on three procedures, and examines the ability of the system to facilitate the design and construction of instructional sequences and materials, the ability of the system to aid the teacher in the instruction process, and the costs and distribution of costs of the system.

Based on an experimental implementation, a number of improvements to the system are suggested. No data were found to show that the system was not feasible. However, cost considerations would seem to prevent immediate small-scale implementation. The system is suggested as a reasonable means of instruction in cases where cost is not a main factor, or where a high degree of control of and feedback on the instructional process is desired. Large-scale implementation did seem to be feasible, but more testing would be required.

By allowing teachers the capability of storing instructional materials and algorithms for instructional sequencing and by providing the facility for ready modification of these, TAIM provides a means for using information feedback to improve the instruction process. By automating the accumulation of large quantities of minute data and by providing the means for quick access to this data for use in determining instructional strategies, TAIM provides a practical means for individualizing instruction. And by providing the means for retrieval of any of the information available and the facility for intervention by the teacher in any of its processes, the system remains under the control of those responsible for instruction.

ACKNOWLEDGEMENT

The writer wishes to express his appreciation to Don Zarsky for his considerable contribution to the design and implementation of TAIM, and also to Dr. Steve Hunka and Dr. Tom Kieren for their counsel.

Funding for this project was supplied by the Division of Educational Research Services and the Department of Secondary Education, and testing was done in the Edmonton Separate School Board's St. Joseph's Composite High School. The enthusiastic participation of the members of the Ed. C.I. 466 class and the teachers at St. Joseph's is gratefully acknowledged.

I wish also to thank my wife Norma, whose continuing support and understanding were sustaining in this venture.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

CHAPTER ONE

INTRODUCTION

Instruction, the business of educators, is a topic of general concern to society. Considerable resources are expended annually on the determination of the <u>content</u> of instruction, the ideas to be taught and learned. Experimentation with the <u>process</u> of instruction, or the way in which the teaching should occur however, is beginning to receive more attention as a topic for analysis.

High on the list of requirements for any new instructional process is a provision for some degree of individualization of instruction. Students have differences. An instructional process which enables the detection of meaningful differences and then provides learning experiences chosen especially to meet the unique requirements of each individual student would probably be effective. But the problems of determining which are the meaningful differences, how they can be measured, and what the corresponding experience should be, remain at best only partly solved.

The topic of this thesis, the Teacher-Authored Instruction Management System, is an experimental process of instruction. It makes use of a computer and falls under the

generic acronym CMI - Computer Managed Instruction. Two computer programs are used which together are called the Teacher-Authored Instruction Manager; the acronym TAIM is used to represent both the process and the computer programs. The system was so named to distinguish its most important advantage over other currently available CMI systems. It is authorable by teachers; hopefully by those in the front lines of the instructional process, those in the classroom.

TAIM was designed by the writer under the following initial assumptions:

1. That the system should be independent of the content or the nature of the instruction which it manages.

2. That instruction can be assembled from textual material and tests, and that algorithmic procedures for determination and assembly of instruction can be determined.

3. That the system should be capable of: assimilating new text, tests, and algorithms and allow modification of existing instructional components; marking tests; keeping records of test results and system transactions; producing printouts for individual students; and providing data on and allowing intervention by teachers in all appropriate aspects of its operation.

4. That, as far as possible, the routine aspects of instruction should be assumed by the management system.

5. That the final responsibility for instruction rests with the classroom teacher and that the system should permit (but not demand) decentralization of control over the instructional process.

The system was implemented on the University of Alberta's IBM /360-67 computer (MTS operating system) by the writer and Mr. Don Zarsky during 1972. Mr. Zarsky, a graduate student in computing science, had the major role in the design and construction of the computer programs.

1. THE PURPOSE OF THIS STUDY

The TAIM System evolved over a period of nearly two years from a vague idea into a complex array of computer routines and instructional plans. The purpose of this study was to develop TAIM to a stage where it could be actually implemented (if only experimentally) and then to assess the feasibility of the system for further use.

Three principal groups are involved in the use of TAIM: the teachers as curriculum authors, the teachers as instructors, and the students. Each of these groups see TAIM from a different perspective. Procedures were used to assess the reactions of the principals as independently as possible and simultaneously, the details cf expenditures were recorded.

2. THE NEED FOR THE STUDY

Before any form of implementation of such a system could occur, it must at least be brought to a prototype stage and tested for feasibility. The TAIM System was designed in an attempt to provide an instructional process which would enable higher utilization of available facilities and improve instruction. This report contains a rationalization for the construction of TAIM in terms of elementary system theory, with a description of how the system works, and data leading to a preliminary feasibility assessment.

3. DEFINITIONS

Throughout this report, a number of terms are used which may not be familiar to the reader. The disciplines of System Theory and Computer Science have their own vocabularies, and as well, certain terms are used in this thesis in a specific and defined sense. The arrangement is alphabetical.

BACKUP: The maintenance of redundant data against the event that the current data will be inadvertently lost or destroyed.

CLARITY COMPLEXITY: Complexity of operation of a system resulting from a lack cf knowledge of how the system's processes take place or the inability to predict exactly how the processing will be done (see Process Complexity).

CMI: Acronym for Computer Managed Instruction. An overall system for educational management in which detailed student information is collected and used for the selection and distribution of curriculum materials (Salisbury, 1971).

COMMAND: A line of text structured according to rules cf syntax which communicates a request to the computer.

DECISION ALGORITHM: A specified procedure detailing the rules for comparison of criteria to result in a decision.

DRIVER: (a) The computer program which assembles student lessons. (b) The series of computer programs which collects statistics, maintains backup, generates and prints student lessons and invokes the MONITOR for offline processing.

ENTER: (a) Signal to the computer that a command or line of data has been completed; issued at a computer terminal by means of the carriage return key. (b) The procedure of typing the command or line of data and signaling completion.

FILE: An organized array of data accessible at any point in the array by computer.

IDENTIFICATION CODE: A string of (usually) six characters beginning with an alphebetic character used to refer to either students or users of the computer system.

INSTRUCTION: (a) The process of making information available in a student's environment. (b) A line of data in the Logic file specifying directions to the computer for assembly of students' lessons.

INTERPRET: The mode of computer operation where the machine proceeds through a series of commands or instructions executing these consecutively.

INTERRUPT: A user in an interractive mode may request an immediate halt to computer operation and indicate that a new command or line of data is to be entered by signalling an interrupt with the special ATTN (Attention) key provided.

<u>KEYWORD</u>: The first word of a command or instruction indicating the category of action to be done by the computer. Also called the VERB.

<u>LESSON</u>: (a) The plan for instruction prepared by a teacher. (b) The printout produced by TAIM for students (also called the PRINTOUT).

<u>MONITOR</u>: The computer program which accepts and executes commands from users.

<u>MTS</u>: Acronym for Michigan Terminal System. The operating system in control of the computer at the Unviersity of Alberta.

<u>NEXT-LESSON POINTER</u>: The label of a logic unit which contains a set of instructions to be executed next for an individual student.

<u>NULL LINE</u>: A line of input to the computer containing no characters; an enter signal alone.

<u>OFFLINE</u>: A mode of computer operation in which a series of commands are executed without continuous monitoring by a user (also called BATCH or BATCH MODE).

<u>ONLINE</u>: A mode of computer operation in which commands are executed as issued by a user (also called INTERACTIVE or INTERACTIVE MODE).

<u>PARAMETER</u>: The second and subsequent words of a command or instruction qualifying the actions of the keyword or verb.

PROCESS COMPLEXITY: Complexity of operation of a system resulting from the quantity of specific incremental processes to be carried out (see Clarity Complexity).

PROMPT: A signal by the computer to the online user that a command or line of data may be entered.

SEMANTICS: The meaning of a command or instruction; the actions requested of the computer.

SYNTAX: The rules for construction of commands or instructions.

USER: A person authorized to enter commands, instructions, and lines of data to the computer.

UNIT: In the Display file, a unit is one complete Display; in the Test file, one complete TEST; and in the Logic file, a unit is one complete logic segment, or a "Day".

4. OVERVIEW OF THE REPORT

This report consists of a rationalization, a description and an initial feasibility assessment of the Teacher-Authored Instruction Management System. Chapter Two contains an introduction to the concepts of system theory applicable to TAIM, and an examination of the possible role of the computer in the instructional process. The major problems of providing individualized instruction are considered and a model of instruction in terms of the flow of information is proposed. Three related CMI systems are discussed.

Chapter Three details the operation of TAIM and the

TAIM System of instruction. The nature and structure of the data kept is presented and the capabilities of the computer processing explained. The more computer-oriented aspects of these topics are discussed by Zarsky (1973). Procedures for use of the system are proposed, and comparisons are drawn with related systems.

Chapter Four outlines three procedures that were used to collect the feasibility data, and the data itself is contained in Chapter Five. The assessment, in Chapter Six, is based on how well the system facilitates the construction of instructional sequences and materials, its capability for management of instruction and distribution of materials, and its cost.

CHAPTER TWO

THE DESIGN AND USE OF INSTRUCTIONAL SYSTEMS

There are a number of perspectives from which to view the instruction process. The growing body of knowledge concerning systems in general is presently receiving attention as a fruitful frame of reference for viewing many procedures, including the process of instruction.

The first section of this chapter begins with an explanation of relevant general system theory concepts and terms. The process of instruction is seen as essentially one of the movement of large quantities of diverse types of information. The position is taken that a computer is a necessary tool for the effective management of this flow. To this end, the relationships between men and machines (computers) are discussed. The section then focuses on instruction as a system and provides a tentative model (in terms of information flow) of the current instructional process.

The problems of individualized instruction are the concern of the next section. These are categorized into four main types:

1. the problem of simply coping with the large amounts of information,
2. the problem of selecting and organizing content for students,
3. the problem of distributing information effectively, and
4. the general problem of the complexity of instruction.

The necessity of a computer as an instructional tool is rationalized by its proposed contributions to the solution of these problems. The main arguments include the advantages of computer accessed disc storage as a media for the maintenance of information, the necessity for presenting information at the appropriate level of specificity, and the ability of computers to support cybernetic control via optimization of information flow through feedback loops. The addition of computers to the process of instruction can enable detailed attention to instructional data that is otherwise not possible. For instruction to be properly tailored for individual students, such attention may be essential.

1. THE SYSTEM CONCEPT

The concept of 'system' or the 'systems approach' has been or is currently being incorporated into almost every field of human endeavor. Historically, the first application was by Von Bertalanffy to biological systems; but the efforts of men such as Ashby, Wiener, and Paynter soon extended the concepts to include other natural physical systems as well as constructions of man (Eckman and Mihajlo, 1965a, p106-7).

Although a system approach may yield the most impressive results when applied to physical science systems, many of the procedures are equally applicable to 'soft' sciences such as psychology and education. In education, almost all system work to date has been done in the area of administration. This is understandable because the concepts are most readily applicable to administrative concerns such as cost analysis, timetabling, and transportation problems. The purpose of this section is twofold: to describe a number of basic systems concepts and to show their applicability to the main business of education - the instruction of students. Of the three following parts, the first discusses system concepts, the second outlines the role of the computer and the third relates system procedures to instruction.

A) SYSTEMS

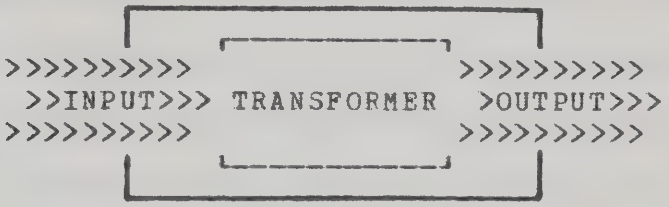
As with most bodies of knowledge, systems theory has an extensive vocabulary. To understand the meaning of the words used is to understand a large portion of the theory of systems. The purpose of this section is to relate systems theory to instruction, and so the pertinent subset of systems vocabulary is herein presented and discussed.

In the words of Banghart: "The term 'systems' defies rigorous definition (1969,p21)." However Webster provides the following: "(A system is) a group of devices or artificial objects or an organization forming a network especially for distributing something or serving a common purpose (1969, p895:1d)." Chorafas, in talking about systems in education states: "A system is a group of interdependent elements acting together to accomplish a predetermined purpose. 'Systems Analysis' is an attempt to define the most feasible, suitable, and acceptable means for accomplishing a given purpose (1965, p2-3)." And according to Flagle: "...a system is an integrated assembly of interacting elements, designed to carry out cooperatively a predetermined function (1960, p58)."

The term 'system' can be used with varying degrees of specificity, making necessary such qualifications as 'sub' system and 'total' system. To illustrate, the human body can be considered as a total system containing a neurological subsystem and a psychological subsystem (to name two). A school system contains many interacting subsystems; as well

as a number of human subsystems people). Whether a system is considered as a total system or a subsystem depends entirely on the focus of discussion.

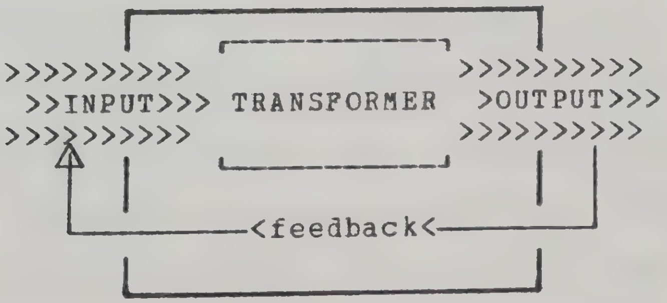
One of the easiest ways to characterize systems is to think of them as transformers; that is, they transform one set of things (INPUT) into another set (OUTPUT) (Andrew and Moir, 1970, p38).



AN OPEN SYSTEM

FIGURE 1

A system is called an open system if it is dependent upon other systems; it has inputs from and outputs to external systems. An example would be an electric light; input is electricity, output is heat and light. Another example would be a school system; very crudely, the inputs are money and uneducated students, and the outputs are educated students.



AN ADAPTIVE SYSTEM

FIGURE 2

Nearly all systems are open systems. Of these, the most interesting are adaptive systems. In an adaptive system, a portion of the output is also an input. The study of adaptive systems is called cybernetics. This word was derived by Wiener (1956, p16) from the Greek word 'kubernetes' or steersman, the same word from which was derived the English word 'governor' (in both the political and mechanical senses). The steerman analogy is appropriate. Considering the steersman as an adaptive system, his inputs are orders from the Captain and the attitude of the boat he is steering. Output is the attitude of the boat caused by his movement of the ship's rudder. The steersman's job is mainly one of monitoring the discrepancy between the boat's current direction and the required course. As he makes changes to the ship's rudder, these are reflected in the boat's attitude. The attitude feedback provides the information necessary to further rudder adjustments. Another often quoted example is the furnace thermostat system. The inputs to (a simple) thermostat are an optimum value and measures of the actual temperature. Output are orders to the furnace to either start or stop producing heat. Of course the results of the furnaces actions are fed back as input to the thermostat in the form of new temperature readings.

Simple adaptive systems behave by monitoring the discrepancy between the values of incoming data and some optimum parameter. Their output consists primarily of signals which can cause the incoming values to be closer to

the optimum value.

More complex adaptive systems are self-organizing systems. This type normally contains several simpler adaptive systems (as subsystems) and by some means determines which of them will become operative. A computer program is a simple example; it may perform an iterative task until some criterion is met and then switch to another task automatically. The inputs, transformations and outputs of these tasks or subsystems need not be the same.

> Much systems work has grown from work in engineering and physics. Two closely related concepts used quite frequently in systems thinking are entropy and randomness. Entropy implies that a system tends to degenerate into disorder. Randomness assumes that any event has an equal opportunity of occurring. Order grows from the random occurrence of events. If a system grows without continuous monitoring, there is a tendency for entropy or disorder to continue. The concept is significant in systems work because the assumption is that one not only designs an effectively operating system but must continue to monitor the system (p33,34). ...
>
> The problem is ... to retain the integrity of the system and to try to minimize the tendency of the system to degenerate (Banghart, 1969, p35).

"The primary contribution of cybernetic theory to systems theory rests with the three concepts of feedback, communication, and control (Banghart, 1969, p35)." Most systems have or can be made to have outputs which would be useful in controlling the system. The problem is one of providing the communication link so that these outputs can be fed back as inputs.

The tendency of any system to entropy is a function of the degree of control available in the system. The larger

and/or more complex the system, the more difficult it is to have proper control; but in the words of Deutch: "Cybernetics is one of the most outstanding methods for dealing with complexity: It respects the vaguely intuitive ideas that we pick up from handling such simple statistics as income or balance of payments data,... and sets to work to a rigorous discipline of sound practical planning (1965, p640)."

Social systems, even small ones, are self-organizing and complex. They are self organizing because they must perform a variety of functions, and they hold the responsibility for deciding which functions will be done at any given time. They are complex because they operate on large quantities of data and usually much of the input information is inaccurate and/or incomplete.

By definition, a self-organizing system has a major function of producing a decision as to which subfunction will operate. This can be called the organization function. In most self-organizing systems, the organization function is complex due primarily to the lack of explicit relationships between system inputs and organization function transformations and outputs. To illustrate: the manager of a small company must make a screening of job applicants. This organizational function is complex because: 1) the manager will be required to screen applicants for widely divers positions, 2) the incoming data from the applicants may not be comparable, and 3) the manager may not

be sure of exactly the type of person that is required. The function is complex not because of intricate or numerous data manipulations, but because the ways and means by which the data should be transformed are not completely clear. This type of complexity will be called Clarity Complexity.

Systems generally have most clarity complexity concentrated in the organizational function. Once a subsystem has been chosen it is just a matter of processing to complete the task. However, the functioning of subsystems can also be complex. Although the inputs, transformations, and outputs are available, the process often operates with a large quantity of data or makes intricate transformations. The processing itself may be routine, but the sheer quantity of operations to be done makes the function complex. This type of complexity will be called Process Complexity. Once a manager hires an employee, he will invoke the routine but complex subfunction of placing him on the payroll.

The two major ways in which a systems approach can aid in the control of a complex system appear to be: 1) the various subsystems can be identified and their interrelationships made explicit so that the results of a subsystem modification can be seen in the context of the total system operation (see Andrew and Moir, 1970, p19), and 2) the existing feedback links can be identified and perhaps improved. In areas where feedback loops appear to be necessary but are not available, they can be designed and implemented.

Advantages of initial system planning are outlined by Elanchard and Cook:

> First, it forces the project participants to simulate the project's operation. This provides a quick and incisive means of reviewing the project as a whole, as well as major portions of it. Third, if it is cooperatively prepared, it represents a negotiated agreement among the various participants to do certain specific things, in specific ways and at specific times. Fourth and finally, once approved and implemented, the plan provides a built-in means for determining progress and problems (1971, p52).

With no attempt to minimize the problems of applying cybernetics to mechanical and electrical systems, it must be recognized that the problem of control of economic, psychological or educational systems is of a different order. Where mechanical/electrical systems can use analog or digital signals to operate cybernetic devices, social systems must depend upon some other form of feedback. Where machines can generate, code, decode, and act upon vast quantities of information at high speed with negligible error, humans generally proceed with varying degrees of deliberation, vacillation and caprice.

There are problems at both ends of the feedback loop; it is difficult to collect the information necessary to effective system monitoring; and then the self-organizing system must identify and retrieve the controlling data from the mass available. It is only recently that attempts to control this type of problem have become feasible. The advent of the computer has singly enabled the application of cybernetic concepts to complex social systems.

B) <u>MAN/MACHINE SYSTEMS</u>

All machines are part of a man-machine system of some sort, because all machines were built by man to some purpose. In some systems the machines operate quite independently, without much human control; clocks, thermostats and automatic city streetlights are examples. Some: the typewriter, shovel, or automobile require constant and direct human control to perform their functions. Complex man-machine systems however require a precarious mixture of human control and automatic machine actions. These systems are most often self-organizing, and as predicted by Edwards: "any man-machine system costing more than $1,000,000 and designed after 1965 will include at least one computer (1962, p76)."

Edwards cites the four main gains due to computers as speed, reliability, precision, and reduced cost.

> It is not just a matter of performing computations in an hour which might otherwise take a week or month; the improvement is much more dramatic than that. Computers routinely do in an hour or two computations which would take an unlimited staff ... with paper and pencil centuries to perform. <u>So computers routinely perform calculations which otherwise would not be performed at all</u>. (emphasis added) ...
>
> No machine is perfectly reliable. But a present-day digital computer, properly used and maintained, may make errors less than once in a billion operations - and if it makes an error, it may be able to detect the fact and make appropriate corrections. ...
>
> Within limits, the computer can carry as many significant figures as are needed through a calculation at no extra cost. ...calculations can be designed for precise though difficult rather

> than imprecise though easy procedures. ...
>
> The savings obtainable from large computers result from the fact that the computer can easily and quickly perform repeated calculations which would otherwise be impossible. In business applications, such computations may save amounts of money literally thousands of times as great as their cost (1962, pp79,80).

Computers can be an asset to social systems in two ways: through mechanization of subsystem processes, and through enhancement of feedback loops. As previously mentioned, many of the subprocesses of a self-organizing system are such that the inputs, transformations, and outputs are known. Even though the subfunction may have process complexity, the knowledge of the nature of the process makes it routine. Computers excell in doing routine work. Further, because of the computer's high speed and reliability, portions of the organization function of a system can be made into subfunctions (of perhaps new subsystems), reducing the overall clarity complexity of the system.

Feedback control loops have two characteristic properties: speed and accuracy of returned information. Generally, the more quickly a controlling mechanism can sense and react to change, the better. However, the optimum accuracy is not always as-accurate-as-possible. Consider the furnace thermostat system. The more quickly the thermostat can sense a temperature change and transmit instructions to the furnace, the more effectively the system will operate. However, a system which can detect only changes of 50^{o} will

not keep a room at optimum temperature. A thermostat which could detect changes of 0.001° would issue so many orders to the furnace that the machinery would soon be worn out, and such accuracy is not really necessary.

Computers can be used to transmit information within social systems. They will tend to be faster than humans at doing this. Computers can also be programmed to use the optimum accuracy. For example, the manager hiring an employee will not need tc know his social insurance number. Yet this information is necessary to the payroll registration subfunction. In a social system, accuracy means that the transmitted information is both necessary and sufficient.

The attributes of speed, reliability, and precision give rise to capabilities of computers to execute some processes better than men. One of these is the ability to search. Suppose a certain class of income tax deduction has been changed. Since payroll records are not normally indexed by tax deductions, a search of all records must be made to determine to which people the new rule applies. A computer can search in seconds a set of records which would occupy clerks for several months. Further, the computer is less likely to make a mistake.

> Another task is which computers excel men is long-term storage. They remember immense amounts of information and can reproduce it extremely accurately. It is seldom desirable for a man to commit something to memory if he can conveniently use a machine. ...
>
> The other ways in which computers excel men

> as technical assistants mostly reflect human imperfections from which computers are free. Among them are: emotional attack on problems, slow speed of learning and necessity for repetition, egotism, and unwillingness to be treated as a slave, laziness, and dishonesty (Edwards, 1962, pp93-94).

Powerful as computers may be, they are unable (nor likely in the future to be able) to resolve clarity complexity except where it can be translated into routine processes. In this area, men are far more proficient than computers, mainly because of their tolerance for ambiquity and their ability to translate uncertainty into probability. Where computers operate only with specific data (which may include human priorities), men are accustomed to translating vague and uncertain information through processes of inference and analogy into bases for action. In an important sense, mean are also more reliable than computers. Machines do exactly what they are 'told' to do; if instructed to proceed in error, a computer will conscientiously repeat the error endlessly until stopped by a human. Where machines must be told what to do under all error conditions, men can both recognize unforseen errors and spontaneously generate corrective courses of action. Finally, men are both cheaper (see Edwards, 1967, p96) and more readily available than computers.

Social systems by definition are operated by men, but most can be made to operate better through the addition of a computer. Functions containing clarity complexity must be handled by men, but it is often possible to delimit rcutine processes in these functions. Similarly, functions

characterized by process complexity can be assumed by a computer after all clarity complexity has been resolved and if that is the better course from both process and economic points of view.

How duties should be divided between men and computers can not be discussed in more detail until the nature and goals of the object system are known. To this end, the following section presents a model of classroom instruction in terms of information flow and attempts to apply the above principles.

C) INSTRUCTIONAL SYSTEMS

Education is a vast enterprise in North America, carried out by various regional and local education systems. These systems can be considered in a hierarchy, each containing one or more simpler subsystems. Currently, in the standard public education hierarchy, the smallest discreet physical system provided for instruction is the classroom. Each classroom has assigned to it a number of students - a class, and at least one teacher.

Each teacher-class pair is the focus of an instructional system. Although an instructional system can be common over several classes, or be under several teachers, at least one class and one teacher must participate. This system is the vehicle by which experiences are organized and prepared by the teacher for his students. The primary inputs to this system are the skill and experience of the teachers; the primary outputs are the

knowledge, skill, and experience imparted to the participating students.

THE OPERATIVE PHASE OF INSTRUCTION

The instructional system is primarily concerned with the flow of information. Although it is necessary to examine how an individual learns when discussing the process of instruction, this topic is considered only briefly. For learning to occur, a student must interact with information available in his environment. The learning process will then be considered as student/environment interaction, and the teaching process to consist of supplying a part of this information to the students' environment. These two processes together are called the operative phase of instruction.

A student interacting with his environment may be considered as a self-organizing complex cybernetic system. Self-organizing because the individual student must determine, at any given moment, which of his subfunctions will operate; complex because this decision (or organizational) process is characterized by clarity complexity; cybernetic because the decision process can be constantly monitored through information feedback. Although the student has a great number of subsystems from which to choose, only four will be explicitly identified. These four appear to be the most frequently engaged, the most observable, and the main subsystems over which the teacher has some degree of control. They are: student/lesson

interaction, student/teacher interaction, student/materials interaction, and student/student interaction. All remaining subsystems are lumped under student/other interaction.

STUDENT/LESSON INTERACTION:

The term 'lesson' is being used in a very specific sense. It refers to the pre-defined procedure chosen or designed by the teacher for producing information in the students' environment. For example, a lesson might consist of a teacher's plan to have a twenty minute lecture on the Number Line followed by a discussion; the content of the lecture (at whatever level of specificity it is available), the plans for the discussion, and any plans for the introduction of materials. It would not include the content of the discussion or any other occurrences not pre-specified or pre-planned. Note that the lesson need not be written down or made explicit before it is used; in many cases it will exist only in the teacher's mind as his intentions when he enters the classroom.

The student/lesson interaction subsystem operates when the lesson information impinges upon the student's environment. In the above example, this information would initially consist of the realization that the teacher was going to lecture, and then the content of the lecture. Each student in this situation may continue to participate in the student/lesson interaction or choose to invoke another subsystem such as: student/teacher interaction (by raising his hand or making a comment), student/materials interaction

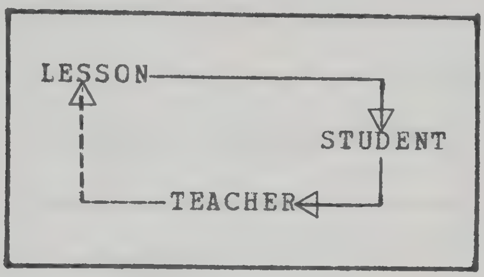
(by taking out a book), student/student interaction (by seeking help from a friend) , or one of the student/other subsystems (such as daydreaming).

The lesson itself may require students to operate subsystems under the student/lesson system. Planning for discussion for example causes student/teacher and/or student/student interaction subsystems to become operative. Planning for the use of materials would invoke student/materials interaction subsystems in most students.

While teaching, the teacher is also acting as a self-organizing system. He may continue with the lesson, or spontaneously devise or choose an alternative lesson. He may also interrupt the lesson temporarily to invoke a student/teacher interaction (if, for example, he notices the student daydreaming). Because of the definition taken, a lesson must be considered as a constant over a given time interval of instruction. The lesson may be interrupted; but if it is altered, it is considered to be a new lesson. The teacher has complete control over these possibilities; student control arises only from their ability to affect the teacher. The flow of information in this system can be diagrammed as shown in Figure 3.

The solid line from the lesson to the student indicates the flow of the content information of the lesson. Similarly, content information on the students' reaction is available to the teacher. The dotted line in this and following diagrams shows the flow of controlling

information; in this case, the teacher controls whether the lesson continues, is interrupted, or abandoned.



STUDENT/LESSON INTERACTION SUBSYSTEM

FIGURE 3

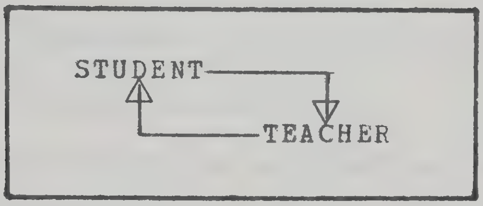
OTHER INTERACTION SUBSYSTEMS:

In the three remaining interaction subsystems, the information flow to and from the student are of roughly equal strength.

The student/teacher interaction subsystem is potentially the most efficient and effective means of instruction available. Here, the student or teacher initiates some information causing a reaction of information generation in the other. This cycle then repeats until one of the participants (or both) stop participating. This is depicted in Figure 4.

The student/teacher interaction subsystem is widely used by students and teachers, mainly to make up for information deficiencies arising from the student/lesson interaction. For example, a student listening to a lecture may raise his hand, causing the teacher to interrupt the lesson. The student then asks for clarification or makes a remark causing the teacher to either ask questions or make

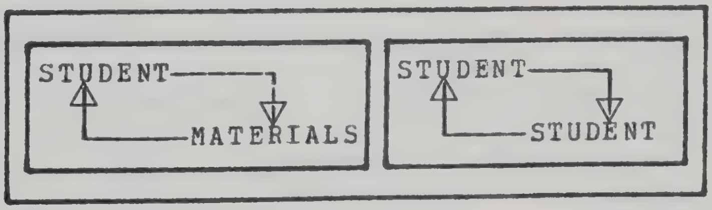
explanations. This information interchange constitutes the interaction subsystem, and will help the teacher identify areas of needed information and supply that information even when it was not provided for in the lesson. In this way, the student/teacher interaction helps to optimize instruction.



STUDENT/TEACHER INTERACTION SUBSYSTEM

FIGURE 4

The student/materials and student/student interaction subsystems are similar. When interacting with materials, there is a flow of content information from the materials tc the student. The manipulation of the materials can be considered as controlling information flowing from the student to the materials.



STUDENT/MATERIALS AND STUDENT/STUDENT

INTERACTION SUBSYSTEMS

FIGURE 5

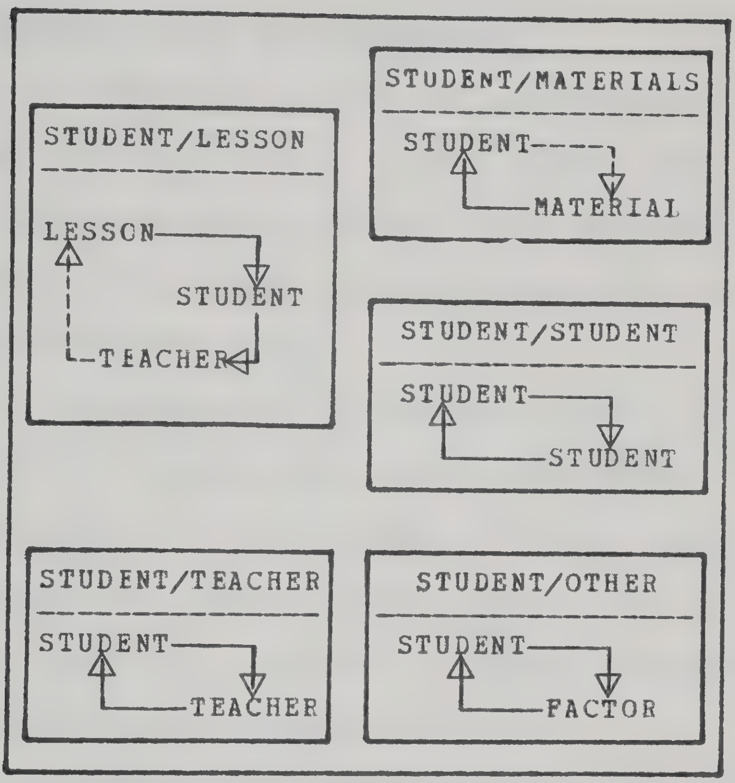
In the operative phase of instruction, the four main interaction subsystems can be operating and interacting in complex ways. The teacher and each student, as self-

organizing systems may have one or more of these subsystems in process at any given time.

The subsystems can be subordinate to themselves or to one another. For example, if a lesson calls for a discussion, a number of student/student and student/teacher interactions can be operating under a student/lesson subsystem. While involved in a student/teacher interaction, the teacher might direct the student to a book causing a subordinate student/materials interaction to be invoked.

Of the student/other interactions, many are important to the instructional process. One of these would be students observing other students operative subsystems. A teacher/student interaction would hopefully generate information in non-participating students' environment. Similarly, observations of other students reading or talking constitutes relevant information which may be a part of some student/other operative subsystem for individual students.

Without attempting to show the complex relations possible between these subsystems, the operative phase of instruction is considered to consist of student/environment interaction. This general interaction contains student/lesson, student/teacher, student/materials, student/student and student/other interaction subsystems.



STUDENT/ENVIRONMENT INTERACTION

FIGURE 6

The inputs to the operative phase of instruction can be named (at the level of specificity of this discussion) as: student readiness, the lesson, and other environmental factors. Student readiness is the information and attitudes each student brings to the class. The lesson consists of the plans for instruction, whatever instructional content is available, and whatever materials have been selected or prepared. Other environmental factors includes such global parameters as the state of the classroom and school morale down to individual physical or emotional concerns. There are also three main outputs: student learning, teacher experience, and other environmental changes.

THE PRE-OPERATIVE PHASE OF INSTRUCTION

Before the operative phase of instruction, there is usually some preparation done by the teacher and perhaps the students. This preparation occurs in the pre-operative phase of instruction. The outputs of this phase form a part of the input to the operative phase.

Student preparation is defined simply as what the student does in preparation for instruction. This may consist of doing an assignment, talking with friends, thinking about the instruction, or nothing at all. Teacher preparation consists entirely of making the lesson (lesson is defined as what the teacher prepares).

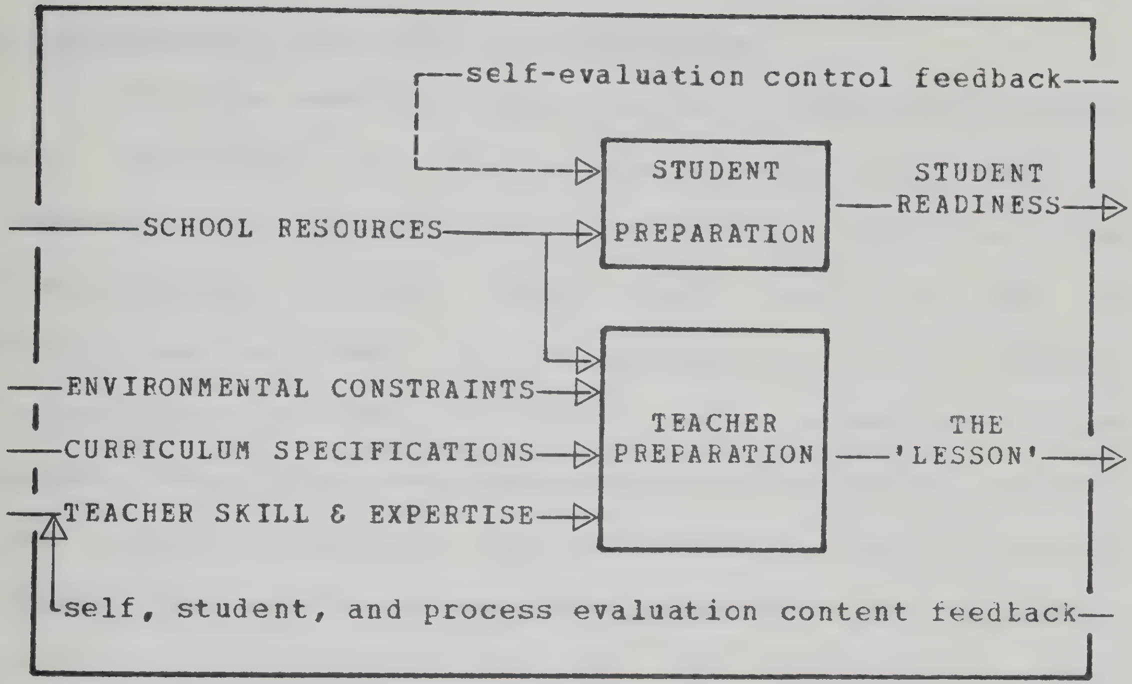
The main inputs to the student preparation process are school resources and control information from self-evaluation. School resources are considered to include any books, games or materials, any school facilities, and any resource personnel available. School resources are also inputs to teacher preparation. As well, the teacher is cognizant of the curriculum specifications and any environmental constraints that might affect lesson construction. Where self-evaluation feedback serves only as control information for the student, self-, student-, and process-evaluation feedback serve as content information for the teacher in lesson preparation. The inputs and outputs of the pre-operative phase are diagrammed in Figure 7.

THE POST-OPERATIVE PHASE OF INSTRUCTION

On completion of the Operative phase, both formative and summative evaluation can take place. One of the outputs of the operative phase was student learning. Both the teacher and student evaluate this, and on certain occasions the student will be given a report on the teacher's evaluation of his learning. Another output was teacher experience. This will consist of information on both student learning and the effectiveness of the operative phase of instruction.

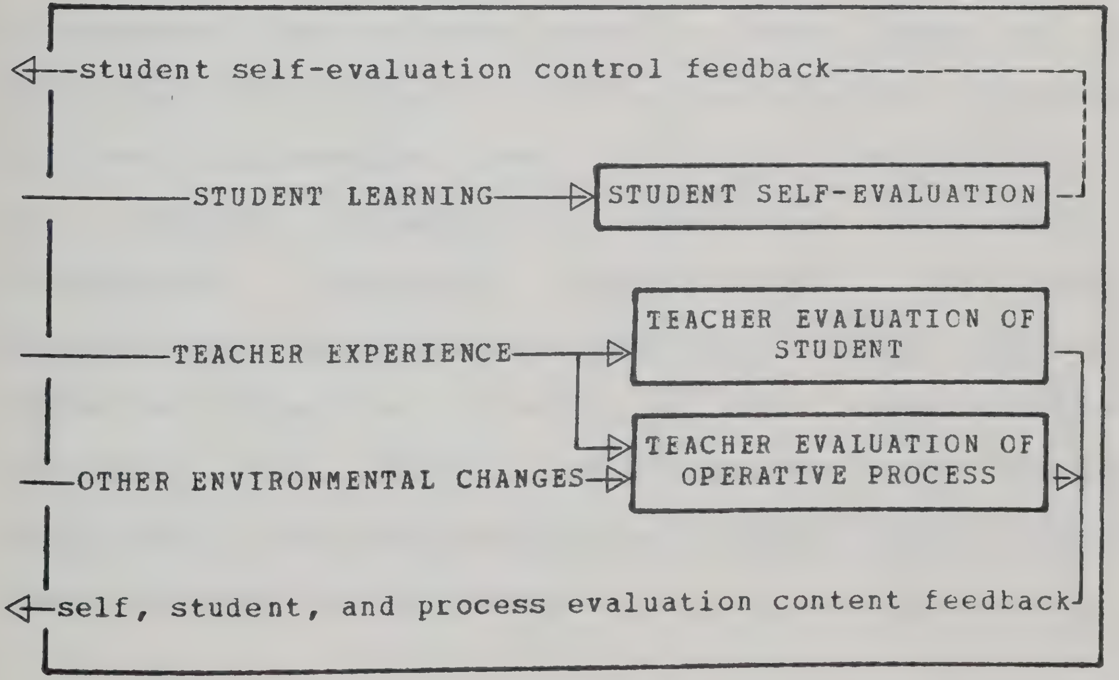
The teacher uses this information to evaluate the individual students as well as the operative phase process. The third source of input to the post-operative phase is information on any other environmental changes caused by the operative phase.

There are two main outputs from the post-operative phase of instruction. Both of these consist of feedback information. The evaluation that the student makes on his own learning has an effect on the amount of preparation that he will do for the next lesson. The teacher's evaluation however, produces feedback information which will be directly used in the construction of the next day's lesson. The post-operative phase inputs and outputs are diagrammed in Figure 8.


self-evaluation control feedback
STUDENT PREPARATION
STUDENT READINESS
SCHOOL RESOURCES
ENVIRONMENTAL CONSTRAINTS
CURRICULUM SPECIFICATIONS
TEACHER SKILL & EXPERTISE
TEACHER PREPARATION
THE 'LESSON'
self, student, and process evaluation content feedback

THE PRE-OPERATIVE PHASE OF INSTRUCTION

FIGURE 7


student self-evaluation control feedback
STUDENT LEARNING
STUDENT SELF-EVALUATION
TEACHER EXPERIENCE
TEACHER EVALUATION OF STUDENT
OTHER ENVIRONMENTAL CHANGES
TEACHER EVALUATION OF OPERATIVE PROCESS
self, student, and process evaluation content feedback

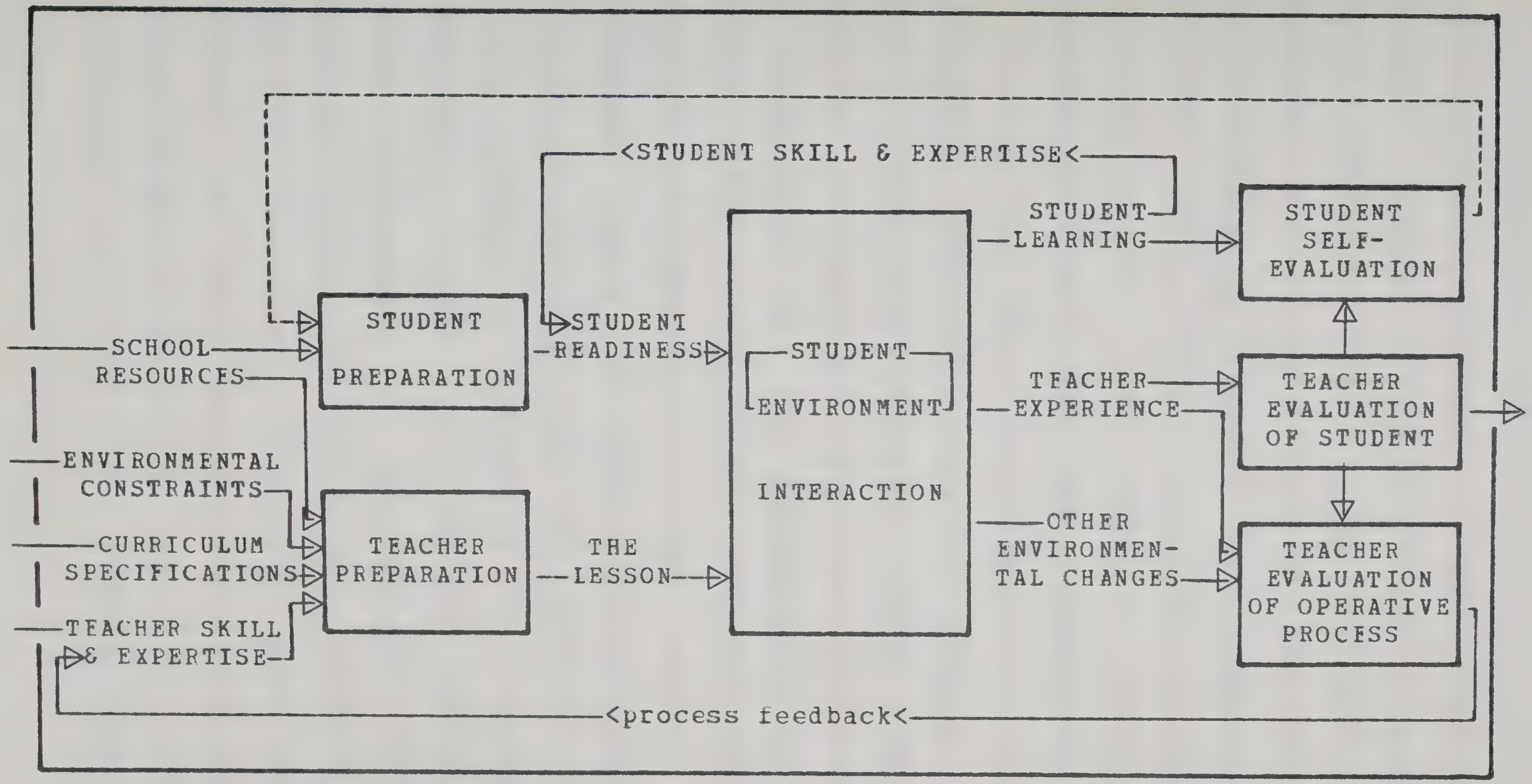
THE POST-OPERATIVE PHASE OF INSTRUCTION

FIGURE 8

AN INFORMATION FLOW MODEL OF INSTRUCTION

The three described phases of the instructional process are collected to form a single pictorial model of instruction in Figure 9. The pre-operative, operative, and post-operative phases must repeat over and over as instruction proceeds. It is natural, in the current instructional setting, to think of a school day as the interval over which one complete cycle of the model occurs. For the loop containing 'student learning', this is perhaps appropriate. However, since the preparation of a lesson requires an iteration of the model, the lesson duration must determine the recursion interval for the loop in which it is contained.

The lines in the diagram represent and only represent information flow; the names given to these lines can be misleading. For example, the lines labelled School Resources represent information taken from school resources and not the resources themselves. Similarly, Environmental Constraints, Curriculum Specifications, and Other Environmental Changes are the names of the storage medium by which the information is kept. In other cases (Student Readiness, the Lesson, Student Learning, and Teacher Experience), the names are of the information itself; and in some accumulated information.


SCHOOL RESOURCES
ENVIRONMENTAL CONSTRAINTS
CURRICULUM SPECIFICATIONS
TEACHER SKILL & EXPERTISE
STUDENT PREPARATION
TEACHER PREPARATION
STUDENT READINESS
THE LESSON
<STUDENT SKILL & EXPERTISE<
STUDENT
ENVIRONMENT
INTERACTION
STUDENT LEARNING
TEACHER EXPERIENCE
OTHER ENVIRONMENTAL CHANGES
STUDENT SELF-EVALUATION
TEACHER EVALUATION OF STUDENT
TEACHER EVALUATION OF OPERATIVE PROCESS
<process feedback<

AN INFORMATION FLOW MODEL OF INSTRUCTION

FIGURE 9

The instructional system is an open system, as shown by external inputs on the left and output on the right of the diagram. The single output shown could be labelled External Reports. This would include report cards and parent teacher interviews where the teacher presents information concerning his evaluation of students. The instructional system may have other outputs such as the teacher's evaluation of the operative process or the student's self-evaluation, but these are peripheral to the instructional system functions.

In the two cases where labels are of accumulated information, these are contained in feedback loops. Student Skill & Expertise is accumulated student learning arising from the operative phase. This accumulation may also occur in systems other than the instructional one. Similarly, Teacher Skill & Expertise is built-up through Teacher Experience with instruction. It can also be extended by information coming from without the instructional system as shown by the unlabelled arrow entering on the left.

As previously mentioned, the purpose of the instructional system is to provide information in the students' environment. In this context, the ability of individual students to select the information they will learn and the amount of information they can learn are relevant only where these effect the amount and type of information made available. The effect is induced through the teacher's perception of the operative process.

INFORMATION STORAGE:

Besides the Student Learning loop (where information is 'stored' in pupils), there are instances where generated or retrieved information is not immediately useful and should therefore be kept by some means until it is required. Perhaps the clearest example of this is the teacher keeping a record of marks students have received. A second example is Teacher Experience which is accumulated as Teacher Skill & Expertise in the teacher's head. In both of these cases, only a fraction of the available information is kept.

In the recording of marks, the teacher is coding his evaluation of student learning over a given time interval into (usually) a single two-digit number. This number may or may not reflect a subjective evaluation, depending on how it was obtained. When the number is subsequently retrieved, it will not reflect any subjective judgement, how it was obtained, nor the circumstances relevant to its determination. This information was available, but was not incorporated into the code that was stored; hence it was lost. This is acceptable only if the information is not needed. Is it important to be able to recall that student X chose wrong answer b) to question 5 of the test on Chapter Six?, that he was absent the day that the topic was covered?, that he is having difficulty two years later learning new concepts to which that question is related?

While the amount of information that can be coded and stored on paper is theoretically unlimited, the amount of

experience that a teacher accumulates is restricted by his selection criteria, coding methods, and storage capacity. Of the mass of data arising from the student/environment interaction, the teacher selects only a small portion as being relevant. The part that is selected must be coded before storage; the teacher is more likely to retain a simple assessment of the operative process ("Things went well with the lesson on the Number Line.") than details of the process itself ("Judy's comment on the possibility of infinite divisions provided a good introduction to the Density concept.") While information kept by mechanical (e.g. pencil and paper) means is relatively stable over long periods of time, an extensive delay between storage and retrieval of information kept mentally normally results in an exponential attenuation.

INFORMATION RETRIEVAL:

The more information that is stored, the greater the problem of retrieval. For example, if all of the tests taken by Student X since grade one had been kept, specific information on his progress in learning a certain concept would be available. Yet it is unlikely that anyone would take the time to sift through all of this data to find the relevant bits. The problem here is partly one of coding (if such tests were kept for all students, ninety per cent of the information would be redundant) and partly one of access (a number of cross-indexing schemes linking related data within students and across students are needed). An

associated problem is that the value of the data is diminished if information as to what was being taught and how it was being taught was not maintained with the tests. An answer given by a student has different meaning if the question was presented long after, just after, or just before the pertinent concept had been taught.

The problem of a teacher retrieving relevant experience from his mind is essentially the same. Without attempting an analysis of how data is stored in and retrieved from a human brain, examples illustrating redundancy, poor access, and lack of supporting data are easily constructed. Where inadequate access methods is the main limiting factor on mechanically stored data, forgetting is the main problem with data stored mentally. Many learning psychologists (Bower, Clark, Lesgold, & Winzenz, 1969; Kintsch, 1968; Murdock, 1968; ... see Kumar, 1971, p. 382) would argue that forgetting occurs partially because the access methods to the information were not constructed or decayed through disuse.

SUMMARY

Instruction is essentially a process of organizing and providing information in the student's environment. Some of this information is input from outside the instructional system, and a great deal is generated from within. Without examining the external inputs, some analysis can be made of the effectiveness and efficiency of use of internal data. A large portion of information generated within an

instructional system could be of great use. The presented model of instruction indicates that only one feedback loop for the storage of operative process data is presently being used; there are means for provision of at least one more. The methods of information storage are inadequate in this age of computers; because of the volume of data that must be processed, only a computer can do an effective job of storage and retrieval of instructional data.

The instructional system is necessarily a cybernetic system. That is; internally generated information is used to control its further functioning. But it is not efficient; due mainly to information attenuation. To function properly, feedback information must contribute to accumulating improvements; the feedback must continually control development so that improvements are not lost through this attenuation. A computer is necessary to an instructional system to store vast amounts of information and then retrieve select portions of it at high speed and with the appropriate accuracy, to search for and summarize relevant bits of data on individuals and groups of students, and to enable the calculations necessary to the mapping of appropriate learning sequences for individual pupils.

2. INDIVIDUALIZING INSTRUCTION: THE ROLE OF THE COMPUTER

That there is great disparity among the abilities and interests of pupils sitting in a common classroom; and that instructional procedures cognizant of these differences would improve the quality and quantity of learning by individuals are taken as self-evident truths in education today. The apparent response to these axioms is to practise what is generically known as individualized instruction (I.I.). Accordingly, a substantial number of techniques for providing I.I. have been devised and implemented. Each of these has associated advantages and drawbacks.

Without examining the pedagogical aspects of I.I., four main instructional management problems can be identified. These are discussed in four following sections: the Coping Problem, the Content Problem, the Distribution Problem, and the Complexity Problem. Adequate solutions to these problems by an instructional system can not ensure that good teaching or good learning will take place. But on the other hand, if these problems are not satisfactorily resolved, the full potential of I.I. cannot be realized. While the majority of I.I. systems are initially designed and subsequently criticized from pedagogical points of view, it is often management criteria that determine their success or failure. Good pedagogy is not possible if poor management techniques prevent proper implementation.

A) <u>THE COPING PROBLEM</u>

An individual can learn in many ways. It is perhaps this flexibility of the learner that has kept traditional forms of teaching viable for the past seventy-five years. The ideal individualized mode of instruction can be realized simply by providing a dedicated knowledgeable teacher for each student. However, it is unlikely that this procedure will be in wide use in the near future; operating student/teacher ratios of between 20:1 and 40:1 will remain as long as cost is a factor in education.

Within the present grade-per-year structure, this means that a teacher could expect to give individual attention to each pupil of each class for a maximum of two minutes per day over the school year - providing he did nothing else. Herein lies the central problem of individualized instruction, for this is clearly not enough time to do proper teaching.

Drumheller (1971) argues that two formats for programs which individualize instruction are available:

> 1. The first implies an ongoing, linear, behavior based instructional program for the major portion of the population. Individual progress is carefully monitored and alternative paths are provided for the individual where the mainstream program has proven ineffective.
>
> 2. The second implies a massive bank of individual learning packages, a massive bank of data on the learner, and a system for matching the needs and learning styles of the learner with the learning packages (p. 34).

These two formats are not as distinct as might appear

on first reading. Whatever the system for matching the learner's needs to the learning units, one of the criteria for needs must be what society expects the child to learn. This along with the fact that a major portion of learning must occur with prerequisite concepts preceeding superior ones (see Gagne, 1970) implies that the learning units of the second format will have numerous linear relationships. Perhaps the main distinction between the two is that the latter contains, in addition to the learning units, both quantities of information on the learners and a "system" for accessing this information in order to match learners with packages. Where the former depends upon the daily monitoring of student progress, the latter has incorporate methods (based on both past and present variables) for determining student-unit matches.

Henderson (1972) notes that the model for I.I. is based on three assumptions:

> First, it is assumed that it is possible to determine for each individual that which is appropriate for him to learn, whether he makes the determination or his teachers make the determination. Second, it is assumed that someone, either the individual or his teachers, can determine the mode or modes of learning that are suitable for him. Third, it is assumed that someone, either the individual or his teachers, can determine an appropriate rate at which his learning should take place. The three assumptions sum up nicely. For an individual it is possible to determine the content, the method, and the rate most appropriate for his learning experiences (p. 17).

From this, Henderson rationalizes that at least one of these assumptions is violated by all I.I. programs and

concludes: "We can best achieve the objectives of individualized instruction by ... (training teachers to) take into consideration the content, method, and pace appropriate for their individual students (p. 22)."

A fourth assumption (not mentioned by Henderson) would be: it is possible for either the teacher or the student to determine an appropriate sequence in which the items of content should be learned. "Following a preplanned sequence, and thus avoiding the omission of prerequisite capabilities along any route to learning, appears to be a highly important procedure to adopt in achieving effectiveness of instruction (Gagne, 1970, p. 243)." Although many students may be able to proceed with a common content or pace, in a common mode or sequence; individualization of instruction implies that the possibility for variance of all of these must be available to each student when this best suits his needs.

Gagne (1970, pp. 329-330) identifies the two methods for design of instruction as extemporaneous design and predesign. Extemporaneous design connotes that the teacher will spontaneously choose the content, mode, and sequence (and perhaps the rate) of instruction for an individual student as required. The alternative, predesign, requires that these choices be made according to preplanned procedures.

> Extemporaneous design of the conditions of instruction during interaction with a student or students, is undoubtedly considered by many teachers one of their most important functions. To

> be done well, it requires a tremendous amount of knowledge and skill, which may be possessed by a certain number of members of the teaching profession, but doubtfully by all. There is no evidence at present as to how well this task of extemporaneous design is done. ...
>
> There is much to be said in favor of predesign of the conditions of instruction. First of all, it appears to be completely feasible. Once the objectives of instruction have been defined, and the learning structure of any topic has been determined, the only remaining action needed is the establishment of the conditions required ... Such a procedure should, if systematically followed, bring about the required learning in the most efficient possible manner.
>
> It may be argued that extemporaneous design can better take into account the individual differences of students. ... But this is an unconvincing argument. First of all, it may be seriously doubted that the typical classroom is the place where the task of designing learning conditions for each individual student can possibly be done. The teacher is simply unable to deal with each student individually in this manner. ... Second, it is perfectly possible for predesigned instruction to take individual differences into account (Gagne, 1970, pp. 330-331).

If the possibility of having each student work on a different portion of content is to exist, content packages must be prepared beforehand. The teacher does not have the temporal resources to provide these extemporaneously. The packages must be such that the student can attend to their contents with a minimum of teacher supervision, and they allow for variability in the rate at which students complete them. Packages covering the same content but couched in different modes of instruction (e.g.: varying media, programmed, group discussion) would also be desirable. A further examination of this is undertaken in the next

section: the Content Problem.

Assuming that content packages are available, a match must be made between each individual student and the optimum content package he requires. Depending upon how often such matches must be made and the extent of the evaluation of criteria for matching, teachers may not have the time or ability to perform this duty either. This problem is discussed under the Distribution Problem. Only if means of predesign of content and distribution methods can be found will the classroom teacher be able to cope with the ranging problems of individualized instruction.

B) <u>THE CONTENT PROBLEM</u>

In the previous section, it was argued that one of the means by which a teacher may attempt to cope with the contingencies demanded by I.I. is to have prepared content packages available. When an individual student need has been identified, one of these packages can be prescribed.

Since these packages will to a large extent determine the content, mode, and rate of instruction for the individual pupils, they must be constructed with some care. Directions as to what content to provide come from two sources. One, in the form of a curriculum guide from an external source dictates the broad areas of content that should be taught. The other, the teachers' perception of the students' capabilities, influences the depth of detail and scope of instruction. While the curriculum guide can be expected to apply to large groups of pupils, variations in

individual capabilities demand a large number of packages; ranging from simple explanations with copious practice activities through detailed analyses and suggestions for further research. Providing these various packages is further complicated in that student capabilities can vary along several dimensions. Some of these are: intelligence, self-sufficiency, tenacity, and a concrete-abstract dimension. An intelligent pupil who is having difficulty working without constant direction will probably require different instruction from one who is intelligent but lacks internal motivation.

One of the ways of providing for these variances is to provide different modes of instruction. Presently, variations in mode are largely determined by available media and teacher training. A "Discovery" session requires training on the part of the teacher that is relatively rare; certain manipulative approaches to mathematical topics require equipment that few schools have available. Only after these external limitations are removed can instructional mode be subject to the proper control - the needs of the students.

There appear to be two main groups who might take the responsibility for the initial preparation of the content packages. In many cases, I.I. program designers have formed or located a group of curriculum experts to do this job. The other possible group would be the classroom teachers. The experts have the advantages of uninterrupted time and above-

average pedagogical expertise. The drawback to this choice is that this group's assessment of student capabilities may be temporally removed and perhaps largely vicarious. The advantages and disadvantages of the classroom teachers doing the initial preparation are the reverse.

Whoever does the initial preparation, the classroom teachers must acquire a thorough understanding of the contents of the packages, for they take the full responsibility of matching them to student needs. They must also take responsibility for recognizing deficiencies in existing packages and the need for additional ones.

The media on which the packages are kept is a crucial factor. Normally printed pages are chosen for their advantages of low cost, flexibility, ease of production, and random access. Disadvantages of this media are that it requires storage space, and once it has been produced it is both difficult and costly to alter. However, these disadvantages apply to most media (film, filmstrips, recording tape, ...) with the exception of random access computer disc storage.

Compared to printed pages, disc storage (and associated computer magnetic tape) is of comparable cost, takes negligible storage space, is easier to produce, has superior random access, and is much simpler to alter. An apparent disadvantage to this media is that information stored there is not directly useable. Content packages on disc would have to be printed out before they could be used. The printing

cost would be less than the production of regular printed pages in the same quantity, but computer printing cannot compare with the quality and flexibility of other forms of information reproduction. Colored pictures or intricate diagrams for example, are not presently possible.

The impact of these disadvantages must be viewed in light of the nature of the content packages. Their main purpose must be to direct student activities so that a minimum of teacher participation is required in the management thereof. This does not necessarily imply that all content material be contained in the packages, for directions on where to find and how to use other media (such as texts, prepared diagrams, films, and resource personnel) are both possible and desirable. Such directions could indicate the mode of instruction in the same way. The packages then might be more appropriately called "direction packages."

When the direction packages are produced, by whatever group, on whatever media, they will not provide optimum instruction nor will they be complete. This statement merely illustrates that no matter how well something is done, there will still be room for improvement. The criteria for improvement of the packages will again come from two sources: the curriculum guide and the teachers' perception of student needs and capabilities. However, unlike the initial production stage, improvements must be most sensitive to the perceptions of the teachers (and

indirectly, the perceptions of the students).

When an I.I. program is produced by a group of curriculum experts, these people recognize the needs for continuous improvement and feedback from the teachers. However they usually retain the responsibility for updating the packages for various reasons including cost and a desire for uniformity. The feedback cycle often employed is to have teachers using the packages prepare and submit written criticisms and suggestions to the experts. They then evaluate these submissions and prepare alterations to the whole program. Of course teachers are not prevented from making direct alterations to the program on their own, but this is discouraged because it causes loss of uniformity and new ideas may not get distributed to other teachers who would use them.

As a controlling feedback loop, this procedure is not efficient. Information is lost both when the teacher encodes (writes) the suggestions and when the expert decodes (reads) them. The amount of information transmitted will be much less than what is available (the teacher will not include all reasons for requesting a modification, nor all suggestions), nor is it fast enough (a suggestion which is accepted may not be implemented for up to a year or more) to provide optimum control.

Clearly the best feedback control will result if the teacher implements his own suggestions directly and immediately. This would result in a loss of uniformity but

this criteria is of doubtful value in a completely individualized program. Some external means of communicating new and profitable modifications to other teachers would then be needed. At this point, further advantages of the computer disc as the media for direction package storage become apparent. Alterations to this media are simple to make and take immediate effect; making the removal of minor irregularities or oversights more feasible. No time is lost in preparation, reproduction, and collating as would be the case with printed paper. Changes would be immediately accessible to all others using disc as a storage media and would take effect for all teachers sharing the same disc with no further action required on their part.

Having several hundred or several thousand direction packages on any media raises problems in itself. These problems center on knowing what is available and being able to find a specific item. Using printed pages, these problems are tackled by tagging each item with a descriptive name or number and preparing cross-references such as a Table of Contents and an Index. Such techniques are also useful when the media is disc. However, since disc can be scanned by computer, programs can be used for locating items using the same type of searching techniques a human would use when looking through a stack of papers. This would be useful if, for example, a new filmstrip became available. The teacher could ask the computer to search for all direction packages where the relevant topic was referenced. Certain of these

which would be improved by a mention of the new film could then be modified. Similarly, if a resource was lost or destroyed, the teacher could quickly find and alter all packages which made reference to it.

The speed and precision of updating techniques available via computer can make feedback control over content or direction packages an important facet of curriculum development. The ability to effect instantaneous changes would enable teachers to do repeated formative evaluations of a specific package in a few days that would take several years or could not be done at all otherwise. And by making changes at a time when students are having difficulty with the directions, the teacher makes maximum use of the available information.

But having a large number of packages available for students does not completely solve the problems of individualized instruction. Assuming these packages were available, the question of how to distribute them to the matching pupils at the right time comes to the fore. And overriding all that has been said, how can any teacher cope with the obvious complexity of I.I., the computer, and all those student differences?

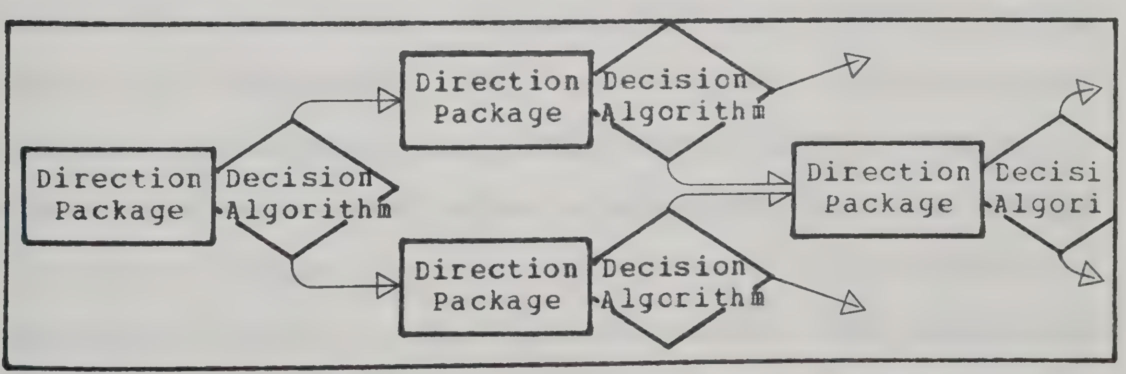
C) THE DISTRIBUTION PROBLEM

With numerous instructional packages that students may use individually available, some means for distributing appropriate packages to students is needed. Ignoring for a moment the mechanical problem of getting the content in the

students' hands; some means of selection, that is choosing the appropriate package for each individual student, is required.

A teacher faced with such a task is likely to devise rules such as: "Whenever a student finishes package A, I will assign package B." Such rules are called 'decision algorithms' because they identify an algorithmic procedure which results in a selection decision. The example algorithm makes use of the criterion of completion of one package for assignment of another. This may be sufficient if package B is the only one available to follow package A; but if other alternatives exist, more discriminating criteria are needed.

Every package will not be linked by an algorithm to every other package. However, increasing the number of alternatives between packages would individualize the sequence of instruction for students in the same way that provision of more packages individualizes content.



LINKING PACKAGES WITH DECISION ALGORITHMS

FIGURE 10

Note in Figure 10 that each decision algorithm is associated with a particular direction package. This seems necessary to expedite location of the algorithms, but does not imply that the distribution criteria used in the algorithm is available only as a result of the associated direction package. Any available information should be useable as criteria (see Graham, 1972, p. 13; and Hunt, 1971).

A teacher faced with the choice of which of two or more directions to send a student may draw on a number of sources of information. These include: the student's general ability, his predicted ability on the task at hand, his preferences for media and mode, his past performance on related or prerequisite tasks; the teachers assessment of the student's needs, and knowledge of the contents and consequences of sending a student in each of the various directions. A preliminary choice must be made as to which of the sources of information to use in determining the criteria for branching. Not all of the possibly useful pieces of information will be available unless planned collection and storage procedures were employed. Requiring a teacher to make such decisions extemporaneously will likely preclude a careful selection of sources; it is doubtful if information would be looked up even if it was collected; resulting in a decision based on a very simple general assessment.

A number of authors (Bloom, 1956, 1968, 1971; Gagne,

1971; Mager, 1961, 1962; Popham, 1969) have taken some trouble to identify and rationalize the need for behavioral objectives and mastery criteria in making such decisions. They would argue that a teacher's extemporaneous summative evaluation of a child is not acceptable as a criterion; that past performance of each individual on related and prerequisite tasks must be measured and these measures used to determine how and what the student should do next. This implies a preplanned set of behavioral objectives for every direction package, and numerous tests to evaluate achievement of these objectives. It further implies an overall planning for learning whereby instructional packages are designed to meet specific needs and are chosen by predesigned algorithms using information that has been planned to be available.

Clearly, pervading and effective instruction management techniques are necessary to keep such a system operating. A number have been designed and tested (for example: I.P.I., PLAN). Many of these employ techniques which vastly simplify various of the above mentioned problems. One procedure is to ensure that all necessary criterion information is collected in the current lesson. That is, branching for each student is determined by information collected during his immediatly previous instruction. A further simplification (for the teacher) requires that each student do his own evaluation and selection of following material (for example: the programmed text; see Crowder, 1960a, 1960b; Mager, 1962), or

provides a completely automatic selection (for example: the teaching machine; see Skinner, 1958; Pressey, 1926).

While these techniques may allow for student preferences and immediate past performance, they ignore the students' general ability, (a top student making a single error could be branched through unnecessary remedial exercises), are inflexible concerning media and mode, have only sequential (or cumulative) access to removed past performance, and tend to prohibit any form of teacher intervention. Most are extremely stable in both content and branching strategies; that is, a teacher noticing a section in need of improvement would have difficulty making changes. In cases where the teacher determines sequence, he is often burdened with the repetitive and boring task of interpreting trivial algorithms. When the students choose their own paths, opportunities for intentional and unintentional miscalculation and ensuing departure from the planned sequence of instruction are frequent. These techniques often result in a mechanistic programming of instruction which has caused a reaction in both students and teachers demanding a more 'humanistic' approach to education (see Landers, 1971; Kaufman, 1972)

Storing decision algorithms on random access computer disc files has a number of advantages, including those associated with the storage of content (as discussed previously) in this manner. The main advantage is that in this form they are accessible to the computer. Since the

computer can search for and find a particular algorithm when it is needed, it would not be necessary to restrict the variety of algorithms; different ones can be provided as necessary at each branch point in the curriculum. Neither is there need to keep the algorithms simple; they can include extensive references to and comparisons of stored data. Finally, since the computer is able to retrieve selected bits of performance data, there is no need to restrict the number or type of variables (criteria measures) included in the algorithms; the computer can look up and use any relevant information necessary for determining a particular students' most appropriate path. Such information might include marks received on a test six months previous, a teacher's recorded subjective evaluation, or the last I.Q. score obtained. In the absence of contrary evidence, it seems reasonable that more powerful algorithms would enable more valid discrimination in the determination of appropriate individual student sequences.

The construction of valid decision algorithms to connect the various direction packages appears to be a monumental task. However, this is not necessarily the case if the algorithms are initially devised at the same time that the content or direction packages are assembled. Each direction package should be constructed only for the purpose of meeting an identified need: either to add a concept to the curriculum or to re-teach a particular topic to individuals who did not attain mastery using the existing

materials. Whatever the reason, the location of this package in the overall curriculum and the criteria for determining which students receive the package must be considered at this time. The specification of the decision algorithm then merely requires that the package author be explicit in his reasons for construction of the package.

Once constructed, the decision algorithms stored on disc files are subject to the same type of repetitive formative evaluation as the direction packages. When, as must happen, students receive a package unsuited to their needs or ability, then either the content of the package was at fault or the algorithm which selected the package did not discriminate properly. In either case, the maximum information would be available to the teacher as soon as the students' difficulty became apparent; he could take corrective action by modifying the package, adding one or more new packages, modifying the decision algorithm, or a combination of any of all of these.

D) THE COMPLEXITY PROBLEM

The task of completely individualizing instruction is certainly a very complex undertaking. Approaches to I.I. which attempt to remove a part of the complexity by removing aspects of individualization or instruction are perhaps useful temporary measures; but they cannot be considered as final solutions to the crucial problem of matching individual pupils' complex needs with appropriate learning experiences. The complexity cannot be ignored or

circumvented if optimal instruction is to be attained. Therefore, means of processing large amounts of information, executing complicated matching strategies and managing extensive banks of data must be utilized. Fortunately, the tools for this task are available; unfortunately, a hundred-fold improvement of instruction may require a doubling of the cost of education.

The first three parts of this section have examined means by which I.I. might be accomplished, the development and maintenance of direction packages (content), and the construction and improvement of decision algorithms. Assuming that a teacher was attempting to individualize instruction through preparing sufficient content-direction packages and the associated decision algorithms, the problem of managing or controlling this complex system remains. It is taken as self-evident that this management cannot occur unless a computer is available to assume a major portion of the routine tasks.

Complexity can be categorized as one of two kinds: 'process complexity' is characteristic of all specified or specifyable operations that could be done by computer; 'clarity complexity' is the property of operations which are complex by virtue of their unspecifyability - operations which must be done by humans. While all educational tasks are initially clarity complex, careful analysis will often enable a translation of a major part of these tasks into a routine process.

There is a certain danger in such a translation; routine processes tend to stabilize over time. Anyone who has become involved in the routine "red-tape" characteristic of governmental institutions must wonder if it is all really necessary. It perhaps was at one time, but those responsible are unwilling to change because they are unsure of the effect on the total system.

> Almost immediately after the creation of a system, defenses designed to maintain a static condition begin to take form. Such defenses often reduce the usefulness of a system in terms of its original purpose. The questions are: how can feedback be provided for a system, and what controls are necessary to assure that the system will be modified on the basis of that feedback? The precise specification of behavioral objectives is one giant step in the right direction. Strangely, a capability that most instructional systems lack is a means for evaluating the product of the system (Knirk, F.G., & Gentry, C.G., 1971, p. 61).

This stabilization can only be countered by constant monitoring and a system view which keeps each subfunction of the total system in proper perspective. Applied to instruction, a total system view can be attained and maintained by keeping a flowchart of all available direction packages connected by the possible decision algorithms. This job may well be done by humans. Although the computer can quickly examine vast amounts of information, it would likely be more difficult to explain to a computer how to recognize proximities among data than to keep track of the relationships externally.

The construction of content direction packages and linking algorithms is a task which must also be done by

humans. The assimilation of various content, pedagogical and psychological considerations is a clarity complex process which may never be done by machine. Fortunately, various current individualization programs are concentrating a great deal of effort on producing such materials.

Finally, it is a human (usually a teacher) who must retain responsibility for instruction of each individual student. This responsibility cannot be taken by a machine; therefore the teacher must monitor both the quality and appropriateness of the instruction being given to each child. The computer can be set to warn a teacher if a majority of students taking a particular lesson are failing an associated test, but it will not be able to provide the reason for this occurrance nor take corrective action by itself. These tasks require the unique abilities of insight and analogy possessed only by humans. The teacher must maintain awareness of the progress of each individual child to ensure that none are abandoned to the processes of the impersonal computer.

The duties of the teacher outlined above are within the range of tasks society expects teachers to perform. The provision of a computer as a teacher helper and the storage of direction packages and decision algorithms on disc files can allow the mechanization of most other tasks. As well, the computer can assist the teacher with the "human" jobs.

> Regardless of the mode of individualization, two factors are quite necessary: continuous evaluation and record keeping. In order to be sure the instruction given each student is commensurate

> with his level of achievement, continuous "checkups" are necessary; in order that each teacher may make the appropriate placement and assignment, she must have some record to which she can refer (Graham, 1972, p14).

If the computer is to be used to interpret the decision algorithms and print out direction packages for each individual student, it can also automatically keep records of which packages each student has received. There must be a close relationship between the content packages given and the evaluation done on the individual students. The computer can distribute tests along with the direction packages and then score the student responses and keep records of these as well. Such tests would include the normal achievement evaluation items, but may also include student preference questions. The scores of these tests would constitute the main source of input to the decision algorithms.

Because of the computers ability to process data at high speed, it may perform calculations and store and retrieve data for students that could otherwise not be done at all. The computer's attributes of accuracy and precision ensure that errors and oversights in the matching of students to direction packages are kept to a minimum.

The responsibility of control of the instructional system is held by the teachers at two levels: development and maintenance of all available instruction and maintenance of proper instruction for each pupil. The computer can help by providing speedy and accurate (relevant) feedback. At the first level, the computer should be able to supply

information such as the location of content packages and tests dealing with certain topics, the "pass" rate of each test subdivided on the possible routes by which students came to take the test, and the probable sequence of instruction for a student with specified characteristics. At the second level, required information would include: a particular student's marks on a specific test or tests, the locations of the last few direction packages he received, and his probable subsequent instruction path. Daily reports on each student's progress would be necessary. To utilize this feedback, the teacher must be able to control the computer and direct individuals or groups of students to new sequences of instruction, place comments or new instructions in students' records, and be able to correct or add new direction packages and decision algorithms. And the computer must allow the teacher this control without requiring that he master a complicated programming language.

Bell (1968, p. 88) notes the problem of a woodlouse in trying to find a suitable damp spot. It has only a single sensing organ and therefore cannot measure a gradient to determine the direction in which greater dampness lay. It's solution is to move randomly; large distances when in a dry area, short distances when the climate is more hospitable.

Innovators in education are often in the same position. Unless they provide means for obtaining measures of relative improvement, they can only sense "good" or "bad" procedures and change their techniques slowly or rapidly as the case

may be. Changes made at random in this way necessarily cause an increase in entropy (see Banghart, 1969, p34) so that the innovation eventually becomes indistinguishable from orthodox teaching practice.

On the other hand, provision of feedback and control permits the creation of order from chaos. That is, one does not need to start with optimum instruction if gradient feedback and directional control are available. One simply (but consistently) improves on the materials at hand: correcting and revising direction packages whenever students have trouble, adding references to new materials or teaching techniques as they become available, modifying decision algorithms to discriminate with more validity and providing evaluation techniques to collect the information required to optimize control.

Just as this basic principle of cybernetics has revolutionized the manufacturing industries, it can be harnessed to transform the educational process that has been so stable for the past hundred years. The advent of the computer has singly enabled this application; needed now is only the inertia to begin the testing and experimentation necessary to determine the most appropriate roles for the computer in education.

3. SELECTED RELATED INSTRUCTIONAL SYSTEMS

There are two main streams of educational endeavor which have bearing on this study: individualization of instruction and the instructional use of computers. The three projects selected for review here are similar to each other and to TAIM in that they bridge these streams. All involve the use of computers for instruction and all involve individualization in the sense that they attempt to supply incremental instruction based on student need assessments. The following three sections discuss: PLAN (Program for Learning in Accordance with Needs), IMS (Instructional Management System), and the Conwell Approach.

A) PROGRAM FOR LEARNING IN ACCORDANCE WITH NEEDS

The project TALENT survey of 1960 administered a two-day battery of tests and questionnaires to about 440,000 students at grades 10, 11, and 12 in the U.S. Flanagan (1971a) notes, among other findings (see also Flanagan, 1971b):

1. A major deficiency in the educational program was the failure to make adequate provision for individual differences.

2. Instructional methods and materials were insufficient to the extent that student performance standards were lower than appear possible.

3. There was a need for a broader focus for educational objectives including those concerning occupational role, citizenship, and leisure.

In an effort to overcome these deficiencies in the educational system, Flanagan established (see Flanagan,

1971b) PLAN (Program for Learning in Accordance with Needs.)

His approach adopted four principal strategies (1971b).

1. To focus the system on the appropriate development of each individual student.

2. To make a comprehensive and systematic study of all of the resources available for assisting students to achieve this development.

3. To place the responsibility for developing the educational system in the hands of a balanced team of qualified specialists.

4. To utilize a management information system to make complete information on each student's progress available to all those involved in both the developmental and operational phases of the system.

Flanagan attempted to implement these strategies through the inclusion of six basic components in PLAN.

1. _A Set of Educational Objectives._ The mastery of each objective was to require two to three hours of learning time, and about five objectives would be included in each "module." In addition, intermediate objectives and long-range objectives were also to be established. These were tested annually.

2. _Learning Methods and Materials._ To assist students in the mastery of their objectives, numerous Teaching-Learning Units (TLU's) were developed. (Examples may be found in Flanagan, 1971a.) There were usually two or more TLUs covering the same objectives, and an effort was made to "match" each student's needs with an appropriate TLU.

3. _Evaluation._ The student was expected to master each of the objectives before going on to the next module. Mastery was assessed by computer scored tests.

4. _Guidance and Individual Planning._ The PLAN system emphasized vocational planning, and vocational information was included wherever appropriate in most courses (especially the Social Studies Program). Through tests and comparisons, with vocational standards, students were counselled in educational planning for the attainment of their vocational goals.

5. Teacher Development. Orientation, Instruction, and Program segments were given to prospective PLAN teachers to help them understand their expected roles. Structured observation of PLAN teachers, computer-supported individualized instruction, and internship were part of this training. Classrocm performance as checked by a consultant with the teacher to insure that all necessary information and skills had been acquired.

6. Computer Functions. The two main functions of PLAN's computer were to monitor students' individual progress and to assist in the planning of individual programs. From a card reader in the school, student responses were read daily and a printout for the teacher produced. This printout contained: a) an exception section noting that new information was required, b) acknowledgements of various teacher actions, c) a planning section, and d) a test-results section (this was usually the main section of the report). Monthly and quarterly progress reports were also produced.

Through a testing and counselling process, the long-range goals of a particular individual were established. Then the student was placed along the scale of the modules in a particular subject for the achievement of these goals based on prior achievement data. Thirdly, a quota was established in terms of the number of modules the student should complete in a year; and finally, the specific modules the student should complete were selected. The student then proceeded towards completion of the modules by receiving TLU's and taking mastery tests on the required objectives.

Early indications of results from PLAN are that:

> The availability of a comprehensive rational approach to individualized education makes it possible for students, teachers and administrators to plan efficient and effective programs of the types demanded by those concerned with accountability in education. ... histories demonstrate very clearly that the availability of this system has made possible important gains in the effectiveness of educational programs for both individuals and groups. The flexibility of such a system also makes it possible to provide a more

> relevant type of individualized education for each student (Flanagan, 1971b, p23).

B) <u>INSTRUCTIONAL MANAGEMENT SYSTEM</u>

System Development Corporation, under a contract from the Southwest Regional Development Laboratory for Educational Research and Development, developed a CMI system called the Instructional Management System (IMS).

> IMS is not a teaching system as such, but rather is designed to aid teachers in managing their regular classroom instruction. IMS serves both diagnostic and prescriptive functions. That is, it helps to monitor every student's performance on behaviorally defined learning objectives, and it also recommends remedial or prescriptive activities for students who demonstrate inadequate mastery of specific objectives. Operationally, this means that each student is tested at frequent intervals, approximately once each week on each subject area; the test results are read into a computer by means of an optical scanning device; and the computer produces performance reports on the following day. ...
>
> Approximately once a week teachers also receive summary reports showing cumulative performance over all tests taken up to that point. In addition, they can request at any time a breakdown of any specified child's performance on designated tests (Coulson, 1971, p162).

During the school year 1969-70, IMS was implemented in two Los Angeles schools. This initial trial enabled the development of learning objectives and criterion-referenced tests. Supplementary exercises were added the following summer; and during 1970-71, IMS was moved to a different school for testing with five first-grade teachers.

> In all three schools each first-grade reading class is divided into three groups: a fast group, a medium group, and a slow group. The teacher typically works face-to-face with one group at any

> given time, while the remaining two groups work with exercise materials or on other activities. Thus the instruction is basically a three-track system, and individualization within a group is limited by the time a teacher can spare to work with individual children (Coulson, 1971, p165).

Three main results were reported by Coulson (1971, p165-166). It was expected that the IMS data would influence the rate at which students progressed through the material. That is, after a group received low marks on a test, the interval to the next test should be longer than when high marks were obtained because of time spent in remedial work. This occurred for only about half of the teachers. For the remainder, group designation appeared to be the main factor determining pacing; 'least capable' groups moving slowly, and 'most capable' groups moving faster. Significantly, pupils of teachers who used the IMS data showed higher gain scores than pupils of teachers who did not.

It was expected that students consistently receiving test scores near the top or bottom of their groups would be reassigned (respectively) to faster or slower groups. This was borne out by the findings; of 94 group transfers, 80 were directly related to performance as reported by IMS.

Thirdly, concerning remedial work; "in about 70 to 80 per cent of the cases, teachers gave the _group_ remedial instruction prescribed by IMS. However, the teachers were much less consistent in administering the _individually_ prescribed exercises, probably because they lacked the time to devote to individual children (p166)."

C) <u>THE CONWELL APPROACH</u>

The Conwell System was designed by American Institutes for Research personnel and implemented in the fall of 1969 at Conwell Middle Magnet School, Philadelphia. As of November 1970, the system was still under development.

Connally (1970) cites four major aspects of the system.

1. The system focused exclusively on basic minimum objectives in various subject matter fields rather than encompassing a complete curriculum in, say mathematics.

2. The system was designed for maximum flexibility in responding to unique sets of objectives defined in the local school setting.

3. There was an effort to assist students in attaining a minimum standard of maturity in a variety of affective and cognitive objectives.

4. The system attempted to adapt instructional treatments to individual differences in children by allowing the student to set his own pace and by sequencing the instructional units in light of student progress.

> In simple terms, the system under development is designed to find out what children need and to supply the most appropriate learning experiences to meet these individual needs. The prototype consists of three basic components: a set of instruments and techniques for assessing student needs, a bank of curriculum packets related to assessed needs, and a computer-based system for relating individual needs to available curriculum options. (Connally, 1970, p3)

Student needs were assessed by measuring students in four ways:

1. Each packet contained a progress test to determine if the contained material had been learned.

2. Four learner characteristics (reading level, aptitude, learning style, and cognitive style) were assessed by means of annual tests. (Published instruments were used except in the case of learning style where an AIR developed test was used.)

3. Before a student started an objective, he was given a pre-test to determine the appropriate "learning segment" (see below).

4. After completing an objective, a post-test was assigned to determine a terminal measure of mastery.

Sixty-seven basic objectives were established in two categories: CONTENT (Communication Arts, Mathematics, Science, and Social Studies) and SKILL (Creative Thinking, Critical Thinking, Communication, Social Behavior, Learning Strategy, and Personel Responsibility). The system focused "exclusively on basic school objectives which are defined as the minimal outcomes of a middle school educational experience (grades 5-8) (Conwell, 1970, p4)."

A bank of approximately 700 curriculum packets were available; as the packets were constructed, they were labelled with a code number to place them in context with the other available packets. Each label consisted of a 14 digit number partitioned into 9 dimensions:

- the first digit was 0 if the packet was designed for use with a SKILL objective, 1 for a CONTENT objective;

- the second digit ranged from 0 to 9 and indicated the area within the dichotomous category above;

- the third and fourth digits specified one of the sixty-seven basic objectives;

- the fifth through eighth digits identified the segiment or specific content under the objectives;

- the ninth digit indicated reading level - 0 (little or no reading), 1 (requiring grade 3 to 5 ability), or 2 (requiring grade 6 or higher ability);

- the tenth digit indicated difficulty level - 0 (simple), 1 (average);

- the eleventh digit indicated learning style - 0 (visual), 1 (auditory), 2 (kinesthetic), 3 (mixed);

- the twelfth digit indicated cognitive style - 0 (abstract), 1 (concrete), 2 (mixed); and

- the last two digits were used to distinguish alternatives between packets with the same categorization on the above eight dimensions.

(Note that this system is capable of uniquely categorizing over 10^{11} packets.)

There are six key decision points in the Conwell Approach instructional process. For an entering student, the computer supplies a list of the unachieved basic objectives. Together, the student and teacher decide which one of these the student will attempt. The student then takes the diagnostic test for that objective; using the resulting information, the computer scans the list of available packets and chooses the one most appropriate.

The algorithm for this choice required first, a search for a perfect match. Failing this (because the packet had not been prepared, or the student had already taken the packet) the computer then searched for an imperfect match following (in order) the rules: allow a lower reading level; allow a lower aptitude; allow mixed learning mode; if abstract presentation was indicated, allow mixed or concrete presentation. If a match was still not found, the computer then ignored the variables (in order): cognitive style, learning style, aptitude level, reading level.

After completing the chosen packet, a progress test was given which the student either passed or failed. If he failed, then with the teacher he decided whether he should repeat the packet, request an alternate packet, or change

objectives. If he passed, then he would either take the terminal test for the objective or request another packet on the same objective. All requests for packets were handled as described in the previous paragraph.

D) SUMMARY

Baker (1971) reviewed the PLAN and IMS systems (among others) and noted:

> They differ ... somewhat in the level of implementation, but the underlying pattern of test scoring, diagnosis, prescription, and reporting is embodied in all of the existing systems. In addition, they are based on a curricular approach in which educational objectives are defined in detail. These objectives then serve as the basis for the design of instructional procedures, materials, measuring instruments, and other aspects of the curriculum (p. 61).

This observation also applies to the Conwell Approach.

While little actual implementation data is available on any of these systems, it appears that the functions performed by the management system are not directly under the control of the classroom teachers. It further seems that the structure of the systems are highly dependent upon the curriculum which they manage. If these conjectures are true, they imply an inefficiency in the curriculum improvement loop and a general tendency towards undesirable stability.

Oettinger (1969) notes three "pitfalls" in the applications of technology to education. The first two of these illustrate the points above.

> First, every attempt to introduce technological change into education has revealed how profoundly ignorant we are. We know precious little about the psychology of learning, and what

> is known is more relevant to the laboratory than to the classroom. Behavioral objectives with a strong taste of the explicit, the quantitative, and the measurable account for but a small fraction of the many effects we expect of schooling (pg. 221).
>
> Second, systems analysts ... are ill-prepared to deal with more than the form of the educational system. ... Teachers ill-prepared and unrewarded even for good teaching in present classrooms are neither well-disposed toward, nor capable of contributing to, technological innovation. Teacher-proof educational schemes prepared by remote academic reformers sit unused on the shelf or are adopted in name only (pg. 222).

The TAIM system, discussed in the next chapter, may avoid at least these pitfalls.

CHAPTER THREE

THE TAIM SYSTEM

The Teacher-Authored Instruction Management (TAIM) system was so named to indicate its major departure from other current CMI projects. It is a system for the management of individualized instruction designed not only to be used, but potentially authored by the classroom teacher.

The purpose of this chapter is to describe TAIM as it would appear to a teacher. (For a more technical description, see Zarsky, 1973.) Because of the interrelatedness of the various components of TAIM, the explanation of any one part in isolation is difficult. Therefore this chapter is arranged beginning with an overall description of the whole system, followed by descriptions of the files and the computer programs. Comments on the design of components are collected in the last section of the chapter.

1. THE STRUCTURE OF TAIM

The rapid expansion of the field of computer applications to education has caused confusion over the various forms of terminology. Salisbury (1971) attempted to establish some guide-lines, and of the definitions he proposed, TAIM best fits his category of Computer Managed Instruction (CMI):

> Computer Managed Instruction (CMI) - An overall system for educational management in which detailed student information (such as that which can be obtained from CAI systems), complete curriculum data and information on available resources are integrated to develop individualized programs of instruction, revise curriculum content, provide for necessary counselling and guidance, and facilitate optimum educational resource management (p. 39).

The purpose of this section is to provide a general description of the nature of TAIM and to establish the setting in which the system would operate. While some elements of a rationale are contained in the following descriptions, a more thorough rationale for the system is contained in part two of Chapter Two of this thesis.

A) OVERVIEW OF THE SYSTEM

The Teacher-Authored Instruction Manager (TAIM) is a set of computer programs which facilitates a Computer Managed Instruction (CMI) environment. Where most CMI systems of instruction available to date restrict and are dependent upon the nature of instruction for which they are used, TAIM may be more properly considered as a teacher

tool. That is; while the communication links between teacher, computer, and student are defined by TAIM, the nature, scope, media, mode, sequence, and rate of instruction are not.

In effect, TAIM is an instructional assistant with the following capabilities:

1. storage of teacher-prepared lessons and tests,

2. storage of teacher-prepared decision algorithms (decision algorithms are the rules by which it is decided which instruction an individual student will receive next),

3. retrieval of specific lessons and tests for individual pupils according to the particular decision algorithms he or she encounters,

4. automatic scoring of multiple choice and integer-value-answer tests,

5. automatic storing of test scores and student sequences,

6. rapid retrieval of specific information on students and across students, and,

7. easy modification of lessons, tests, decision algorithms, and the sequence of individual students.

In designing TAIM, an attempt was made to separate all instructional tasks into those which could be done by a computer and those which could not. Computers cannot create or revise curriculum materials, choose or invent strategies for material presentation, nor can they assume the responsibility for completeness and accuracy of curriculum or the education of children. These tasks and responsibility must be taken by humans; specifically, teachers. However, under human guidance TAIM will store curriculum content, tests, and instructional strategies; execute the

instructional strategies to distribute content and tests to the appropriate pupils; score tests and store the results for use as criteria in instructional strategies; and allow the teacher to retrieve selected parts of the stored data and modify content, tests and strategies when necessary.

The majority of a teacher's duties are complex; but of these, a good portion are such that algorithms can be devised for their execution. The computer is not capable of devising algorithms, nor of ensuring that they are working properly - these jobs must be done by a human. But the computer is naturally more adept and efficient at executing complex processes that have been prespecified. In an individualized setting where the teacher's time is at a premium, it is clearly useful to free him for tasks which he alone can do.

There are three broad categories of tasks that fall to a teacher using TAIM:

1. devising and revising instructional material and algorithm rules for the computer to execute,

2. monitoring the actions of the computer and the reactions of the students to ensure that instruction is proceeding effectively,

3. giving individualized personal aid to pupils when their instruction as supplied by the computer has failed, diagnosing the reason for failure, and making the appropriate revisions to the material and/or algorithms.

The students using TAIM receive, at the beginning of each class period, a computer printout detailing their activities for that class. They also receive a response card. The printout contains:

1) The student's name and the name of the class,

2) The name of the current and previous Logic units,

3) The scores they received on last day's test (if there was one), and

4) The Displays and Tests as determined by their current Logic unit.

The Displays direct their activities by providing instructions, textual presentations, and exercise questions or problems. Students who receive tests mark their responses on their response cards. At the end of the class all students turn in their cards and these are input to the computer.

While there is one set of instructional materials for each subject, students follow different paths through the total curriculum. The path they follow and the materials they receive are controlled by the two parts of each unit in the Logic file. The parts are executed in two steps.

Each student has a "pointer" indicating the logic-unit or "Day" he is to receive next. In the first step, this Day is looked up in the Logic file and the lesson construction statements found there are executed. The student receives his lesson printout, does the indicated work, and turns in his computer card. If the lesson contained a Test, the student pencils his responses on this card.

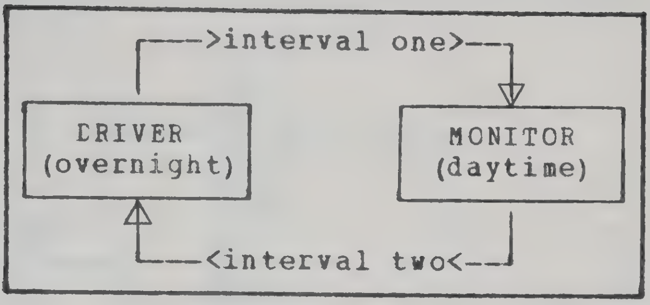
Step two occurs when the card is read by TAIM. If a test was given, the student's responses are marked and the results stored in his Student Record; then the decision algorithm part of the Day is executed and the student's

next-lesson pointer set accordingly.

Although these steps are closely related, they are executed at different times by different programs. Step one occurs during an overnight offline program run of the DRIVER. When this program starts, all students' next-lesson pointers are determined and available. Lessons are printed for students on the high speed printer through interpreting the necessary lesson construction statements. When finished, the program leaves all students' next-lesson pointers undetermined (that is, not pointing to a next lesson), but keeps a record of the algorithm statements to be processed to determine the pointer for each student.

The next day, the bundle of lessons is delivered to the school and distributed among the students. Each student is responsible for doing the assigned work and preparing a computer card. The collected cards are input to TAIM; and the MONITOR marks all necessary Tests and then interprets each student's decision algorithm statements to set their next-lesson pointer. That evening, the overnight program runs again and the cycle repeats.

This cycle is depicted in Figure 11. During both intervals one and two, the teacher may execute other online commands of the TAIM system. These commands are described in a following section; they allow the teacher to gather information about student performance and the contents of students' lessons, and to alter student's next-lesson pointers.



THE MONITOR/DRIVER CYCLE

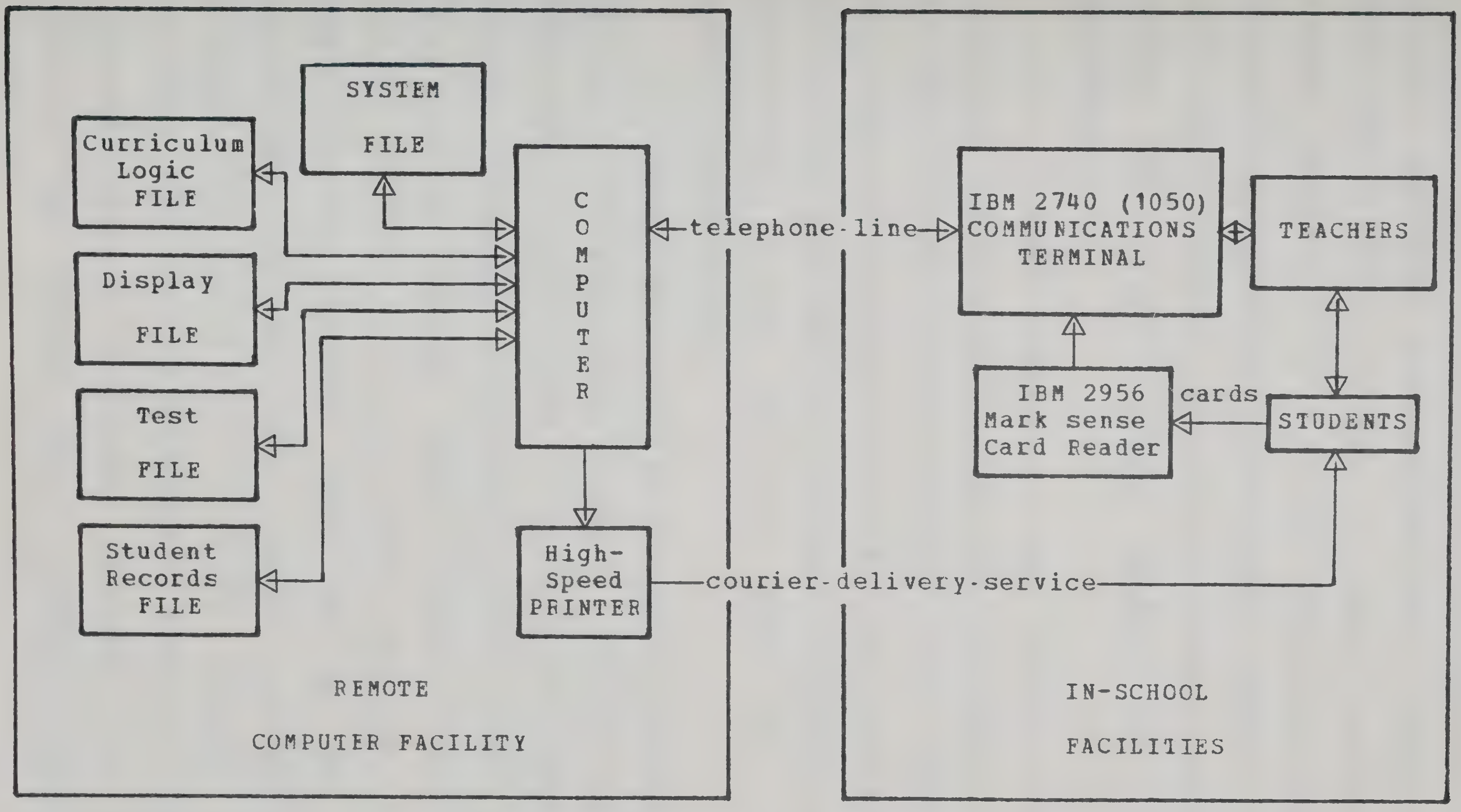
FIGURE 11

B) <u>HARDWARE COMPONENTS</u>

One of the problems of trying to use computers in the instructional process is that most schools have little or no computer hardware. Neither do they have the funds to acquire such devices. This is perhaps the most important single factor keeping CAI in the experimental stages while CMI is able to move into the schools.

The minimum in-school equipment required for the operation of TAIM is a single computer terminal and a data set. However, the design of TAIM also calls for an optical mark card reader (OMR) to read-in student response cards. If an OMR was not available, the student responses would have to be entered at the terminal.

While in-school computer hardware is scarce, most schools are close enough to a central computer facility to make access possible. Almost any such facility would have the remaining hardware components required by TAIM: random-access (disc) storage and a high speed printer.



HARDWARE CONFIGURATION AND COMMUNICATICN LINKS

FIGURE 12

Figure 12 shows the designed configuration of hardware and the necessary communication links. A normal telephone line would be used to connect the school terminal to the central computer. Via this line, the teachers would input curriculum material and modifications, student responses, and systems control commands; and receive confirmation of actions taken, and various student and curriculum information.

The student printouts and certain student and curriculum information printouts would normally be too large for display on the in-school terminal. These would instead be printed on the central facility's high speed printer and delivered to the school by a courier. The courier would make this delivery at least once each school day, preferably before classes started.

A computer the size of an IBM /360-67 would be capable of running TAIM for a large number of schools (say 50 to 100). While telephone links are available to practically all schools, providing long-distance courier service might not be as feasible. However, remote stations linked to the central facility by larger volume telephone lines is a viable recourse currently being used by many business applications. One such remote station in a school district or county would likely prove the most economic solution. Printouts could then be delivered by school bus. By routing in-school terminal connections through this station, telephone line charges would be reduced. Such a facility

would have further benefit to the community, providing the power of a large computer to local and regional government offices and perhaps to local businessmen.

While the designed system configuration does not include computer tape facilities, these would be necessary, at the central facility, for the operation of TAIM on any large scale. Random-access is required at times for each of the files, but some operations require only sequential access. When random-access is required it usually references only a portion of the total file. Since tape storage is much cheaper, large files would be more economically kept on tape when not in use. When random-access was required, only the necessary portions of the file would be placed on disc.

C) THE MAJOR FILES

The main software components of the TAIM system are two computer programs (the MONITOR and the DRIVER) and the main files. The MONITOR and DRIVER are described in following sections. Their main task is to transmit information between the files and students, between the files and teachers, and within the various files.

TAIM keeps about ten information files on random access disc storage attached to the computer. Of these, four are central to the operation of the system: the Display file, the Test file, the Records file, and the Logic file. Each of these files is partitioned into a variable number of 'units'. Each unit is labelled so that it may be referred to by both the TAIM user and the computer programs. Units of

different files may have the same label; where confusion might arise, the labels are prefixed with D- (for the Display file), T- (for the Test file), R- (for the Records file), or L- (for the Logic file).

TAIM keeps one set of these files for each subject group using the system. However, as individual students meet or do not meet the criteria set for them by the teachers, they are channeled to different portions of these files. In a sense, the Logic, Display, and Test files specify a curriculum for the course; individual students proceed at different rates and respond differently so that they access different parts of the total course specification. The Records file keeps track of their past and present positions, and their Test results.

The Display file has the simplest structure of the four. Each labelled unit is called a Display and consists of a variable number of lines of textual material. Displays are printed out for students on a high-speed printer by the DRIVER. The contents of Displays may range from short instructions to a student (to read a portion of a textbook, view a filmstrip, meet with a resource person, etc.) to lengthy discussions of concepts. In short, Displays consist of any printed material that a teacher might wish to be given to an individual student.

The Test file contains a variable number of labelled units called Tests. Each Test has two parts: a textual part and a 'weights' part. The textual portion is similar to a

Display and is the part printed out for students. This part would normally contain a number of test questions and instructions for the recording of responses to these. Since the responses must be capable of input to and analysis by the computer, the questions are restricted to either multiple choice or integer-value-answer variety. The Weights part of each Test contains coded information for the marking of student responses, and so constitutes an answer key to the Test.

The Records file contains a unit for each student registered for the course. Each unit is called a Student Record and is labelled with a student's identification such as the the first five letters of his last name followed by two initials. These records contain registration information (such as the student's age, sex, grade, home room number, etc.) and a complete record of all lessons (Displays and Tests) he has taken, the marks for all Tests, and a pointer indicating which lesson he is to be given next. (It also contains NOTE and HOLD information discussed in section two of this chapter.)

The Logic file contains a variable number of labelled units called Days. Each Day consists of two parts: a lesson part and an algorithm part. Each part contains coded statements which control the assembly of Displays and Tests into individual printouts or specify the decision algorithms; each coded statement consists of a keyword followed by a number of parameters. When constructing a

lesson for an individual student, TAIM proceeds by writing the student's name at the top of a page, locating the particular Day that the student is at, and then executing the coded statements sequentially. The possible coded statements for the lesson construction are:

SHOW D-label causes a printout of the content of the Display labelled "D-label" (in the Display file) to be made.

TEST T-label causes a printout of the textual portion of the Test labelled "T-label (in the Test file) to be made.

MESG "message" causes the student's identification, the date and the "message" to be queued for the teacher. By placing these appropriately the teacher can cause the system to warn him of occurrences requiring his attention. (For example: a student entering a difficult section or requiring special assistance.)

IF (condition) SHOW D-label, or

IF (condition) MESG "message" causes a "condition" (explained below) to be evaluated. If the condition is true, then the SHOW or MESG instruction is executed. Otherwise it is not.

The second part of each Logic unit contains the decision algorithm specifications. There are only two acceptable statements:

WHEN (condition) GOTO L-label and

GOTO L-label

The WHEN instruction allows the GOTO instruction to be accepted only if the "condition" is true. The GOTO instruction simply sets the students next-logic-unit pointer to the specified L-label. These statements are used to determine which Day a student will be assigned to when he has finished the current one. The statements are examined sequentially (for each student) until the next lesson

pointer is set by a WHEN statement with a True Condition or by a GOTO statement. Each Day must have exactly one GOTO statement which is last in the unit. There may be any number of WHEN statements.

The conditions in the above IF and WHEN instructions enable the testing and conditional actions based on two types of information: test results and previous student assignments. Conditions are always enclosed in parenthesis and take the form:

(label R value)

Where "label" is either the label of a Logic or Test unit, "R" is a relationship and "value" is an integer or percentage value. If the label is an L-label, then it represents the number of times that the student encountering the condition has been assigned that Logic unit; if a T-label, then it represents the student's score on the named test. The relationship, R, may be EQ (equal), LE (less than or equal), GE (greater than or equal), NE (not equal), LT (less than), or GT (greater than). "Value" is a positive integer value followed by an optional per cent symbol. If the label is an L-label, then the per cent symbol is not permitted.

Consider the condition (L-Day2 EQ 2). For each student who encountered this condition, TAIM would compare the number of times he had been assigned the lesson controlled by Logic unit L-DAY2 to the value 2. If the number of times was equal to 2 then the condition would be true; otherwise

it would be false.

Consider the condition (T-TEST2 GE 8). For each student encountering this condition, TAIM would compare his total score on that test to the value 8. The condition would be true only if his total score was greater than or equal to 8.

Consider the condition (T-TESTB LT 50%). In this case, TAIM would calculate the student's percentage score on Test TESTB and set the condition true if the result was less than 50%.

Two conditions may be tested simultaneously by placing a logical operator AND or OR between them. For example

(L-DAY2 EQ 1 OR T-TEST5 GT 50%)

would be true if either the student encountering this condition had been assigned L-DAY2 once before, or he had received more than fifty per cent on Test TEST5, or if both of these were true.

A further extension to the condition was planned but not implemented. It would have allowed the appending to Test labels of specific question numbers. For example,

(T-TEST1(3,5,7) EQ 3)

would have been true only if the sum of the student's marks on questions three, five and seven of TEST1 equalled 3. This extension would also have permitted the testing of students responses (as opposed to their scores) on specific question. For example, (T-TEST2(1) EQ 3R)

would have been true only if the student had given a response of 3 to question one of TEST2. This might or might

not have been the "correct" answer to the question.

There are clearly a variety of possible extensions to the described Conditions. However, the means of constructing Tests and inputting data make the possible T-label values quite flexible. For example, if a teacher wished to discriminate on student I.Q., he could make up a single question test, label it 'T-IQ', and then code and read-in cards with each student's actual I.Q. on them. Afterwards, any conditional statement such as IF(T-IQ LE 110)SHOW REV3.3 would be processed appropriately.

2. THE ONLINE INTERACTIVE MONITOR

As has been mentioned, there are two major computer programs under TAIM: the MONITOR and the DRIVER. The MONITOR is that part of TAIM through which the teacher constructs curriculum and directs the instruction of pupils.

A) AN OVERVIEW OF THE MONITOR

The MONITOR actually consists of three hierarchical subprograms, called Mode One, Mode Two, and Mode Three. Using Mode One, the teacher may collect and print out data on the students and the four main files, and may alter the sequence of instruction for individuals or groups of students. The Mode Two program enables the teacher to modify the Logic, Display and Test files. The separation is a natural one; Mode One is used to run the TAIM system, and Mode Two to alter its operation. The Mode Three programs consist of caretaking and initialization routines. Their use

requires an understanding of the architecture of TAIM; they will not normally be accessible to the teachers.

A <u>syntax notation</u> is used in this section for specifying the syntax of the TAIM MONITOR commands. The rules for this notation are as follows.

1. Every command consists of a KEYWORD followed by up to three PARAMETERS. The parameters are separated from the keyword and from each other by at least one blank.

2. If a keyword or parameter is in upper case, then the text of that keyword or parameter represents itself. If a portion of this upper case text is underlined, then the whole keyword or parameter may be abbreviated by the underlined portion.

3. A parameter specified in lower case must be replaced by the user with an appropriate value or string.

4. If several alternatives are allowed for a parameter, the alternatives are listed, separated by OR symbols "|" and enclosed in brackets or braces.

5. If a list of alternatives for a parameter are enclosed in braces "{...}", then exactly one of the alternatives must be chosen. If a parameter or a list of alternative parameters are enclosed in square brackets "[...]", then the parameter is optional; if omitted, it may take a default value.

The invoking of TAIM and three TAIM commands are common over the three modes. In order to use TAIM, the teacher must be registered to the system by a Mode Three user. At this time he is given a six character identification code. As soon as the system has been started, it asks for this code; and will not proceed until it has been properly entered. The system then automatically enters the highest mode to which the user is authorized.

When ready to accept a command, the system types "READY" and waits for the user to enter one. If the system

is able to execute the command, it does so and responds with "DONE ... $$$" where "$$$" is the cost of executing the command. Between the issuance of a command and its completion, the user may interrupt execution via the attention key. If he does so, TAIM will allow the user to have the command queued for re-execution by the MONITOR at the time when the DRIVER is run. This is useful when the command causes a large amount of printing which is not needed immediately.

Three TAIM commands are available in every mode:

<u>MODE</u> { 1 | 2 | 3 | ? }

The MODE command has a single parameter which may be 1, 2, 3, or ?. The keyword, MODE, may not be abbreviated. A parameter of "1", "2", or "3" will cause a change of modes if the user is authorized to enter the requested mode. If the parameter is "?", TAIM prints the value of the current mode.

```
Examples:     a) MODE 1
              b) Mode ?
```

<u>W</u>HY

The WHY command has no parameters, and may be abbreviated by a single "W". There are numerous reasons for a command issued to TAIM not being executed. Whenever this happens, the system responds with "NOT DONE ... ##". The "##" is the number of an error message which explains why the command was not executed. Issuance of the WHY command causes this message to be retreived and printed.

EXAMPLES: a) why
b) w

QUIT

The QUIT command has no parameters. Its issuance simply terminates the TAIM monitor.

B) MODE THREE

In any system as complex as TAIM, there are certain operations which must be carried out only by persons sufficiently knowledgeable about the whole system to fully understand the consequences of their actions. Commands of this category are provided in Mode Three where only the highest level users have access to them. Besides MODE, WHY, and QUIT, six further commands were implemented in Mode Three.

REGISTER userid

This command allows the Mode Three user to register any other user to the TAIM system. The "userid" is the six character identification code assigned to the new user. On issuing this command, TAIM requests the users full name and his authorization level (1, 2 or 3). If the authorization level is 3, then the new user may enter any mode; if 2, he may enter either Mode One or Mode Two; and if 1, he may use TAIM only in Mode One.

UNREGISTER userid

The unregister command causes TAIM to print the user's name and request a confirmation. If the confirmation is given, then that user's registration is deleted from the list of

users.

SET ## {ACTIVE | INACTIVE}

Every command in TAIM has a status of either ACTIVE or INACTIVE. The Mode Three user may set this status with the SET command. "##" is the number of the command whose status is to be changed (this number may be determined from the DISPLAY command below).

```
EXAMPLES:     a) set 8 inactive
              b) set 42 active
```

DISPLAY {USERS | COMMANDS [1|2|3]}

If the first parameter of this command is "USERS", then a list of the currently registered user IDs, their registration dates, the date of their most recent use of the systems, a count of the number of uses, and their full names is printed. If the first parameter is "COMMANDS" then a list of commands with command number and status is produced. If the second parameter is 1, 2, or 3 then only the commands for that mode are displayed; if the second parameter is omitted, then all commands are displayed.

OFFLINE

The OFFLINE command allows the Mode Three user to place MONITOR commands in a command queue. At the time when the DRIVER program is run, the MONITOR is also invoked to process commands queued by execution interrupts or OFFLINE commands.

INITIALIZE {x-FILE|ALL}

This command is used to cause certain or all TAIM files to be emptied of their contents and then conditioned to accept the data they are to hold. If the parameter is ALL, then all TAIM files are so intialized (or re-initialized). If the parameter is "x-FILE", then x must be replaced as follows:

- L to initialize the LOGIC file
- T to initialize the TEST file
- D to initialize the DISPLAY file
- S to initialize the STUDENT RECORDS file
- H to initialize the HOLD file (see Mode One HOLD command)
- N to initialize the NOTE file (see Mode One NOTE command)
- U to initialize the USERS file

```
EXAMPLES:     a) init all
              b) init h-file
```

MTS [mts-command]

This command will allow the user to pass an MTS command to the supervisor system for execution. If the command is given as a parameter, then only the one command is executed. If no parameter is specified then TAIM will accept MTS commands until a NULL (no character) line is entered. If a RUN, LOAD, or UNLOAD command is passed to MTS, the TAIM system will abnormally terminate.

C) <u>MODE TWO</u>

Mode two would not normally be accessible to all teachers. This is because the modification of the Display, Test, and Logic files must be done very carefully - by someone who has a good grasp of the mechanics of the TAIM files.

When making up a Day for the Logic file, the teacher should typically think in terms of one interval of instruction for a student. The duration of this interval may vary for different applications; one school day would be appropriate for regular classroom instruction, but bi-weekly to monthly intervals may be better for adult education programs. For every interval, a presentation must be made; the student must be given some content. The teacher can examine the existing Displays to see if there is one or more available to suit his needs. If additional Displays, such as further explanation, directions for supplementary material, or a set of exercises are needed, these must be constructed, labelled, and inserted into the Display file. The teacher may decide that a quiz on the content is appropriate. If one is available, it can be used; otherwise a Test must be constructed, labelled, and inserted into the Test file.

With the necessary Displays and Test available, the teacher can begin construction of the Day with SHOW, TEST, and MESG statements. The DAY must have one or more exits to other Days which already exist. When these are chosen and the Conditions defined, the WHEN and GOTO statements can be

placed at the end of the Day under construction. This Day must also have an entry; that is, one or more other Days must have possible exits to it. These must be set up by modifying one or more of the existing Days.

Care must be taken to ensure that all Days have both entries and exists, and that all units are properly labelled and contain meaningful statements. Failure in this regard could cause a number of students not to receive their lessons, or to receive improper lessons. However, Mode Two has a number of built-in checks to help eliminate these problems.

The Mode Two user is given three temporary workspace where he can construct any units he wishes. He may place anything at all in these, but the only way to have them become part of the permanent files is through the SAVE command. This command runs a thorough check on all syntax.

As was explained previously, there are four main TAIM files:

1) The LOGIC file contains a number of units called "days". Each Day has a unique label and contains coded statements which determine the sequence of instruction for individual pupils. The statements allowed in the LOGIC file are:

```
SHOW Display-label
TEST test-label
MESG "message-to-the-teacher"
IF   (condition) {SHOW Display-label | MESG "message" }
WHEN (condition) GOTO logic-label
GOTO logic-label
```

2) The DISPLAY file contains a number of units called "Displays". Each Display has a unique label and contains textual material to be presented to students.

3) The TEST file contains a number of units called "tests". Each Test has a unique label and contains both the text of the Test and a test KEY. The Test text is given to students when they are asked to write the test, and the test KEY is used to score their responses.

4) The RECORD file contains a number of units called "student records". Each Student Record belongs to a student registered on TAIM, and contains information as to which Days he has "taken", which Displays he has "seen", and which Tests he has taken along with the marks he received.

All file operations are accomplished through one of the three workspaces called WS1, WS2, and WS3. In order to SAVE a unit in one of the three files, the MODE 2 user first constructs the unit in one of the three workspaces and then asks TAIM to SAVE the unit in the appropriate file. TAIM always checks to see that the new unit meets certain standards before saving it. Similarly, certain conditions must be met before TAIM will allow the modification or DROPping of an existing unit. A short version of the restrictions is as follows:

1) Every unit has at least one line of text.

2) Every unit has a label.

3) Every label starts with a letter and is between 1 and 20 characters in length.

4) Every LOGIC unit contains only acceptable LOGIC statements, and ends with a GOTO.

5) Every label mentioned in the LOGIC file must exist in one of the three files.

6) Every TEST unit has a test KEY (see the SETKEY command).

In the following command descriptions, two abbreviations are used. Five of the commands require the name of a workspace as one of the parameters. This name may

be WS1, WS2, or WS3; and these three alternatives are expressed as "ws#". That is;

```
┌──────────────────────────────────────────────────┐
|    ws#  is equivalent to  { WS1 | WS2 | WS3 }    |
└──────────────────────────────────────────────────┘
```

Four of the Mode Two commands require references to unit labels. Every unit label consists of a prefix {L- | D- | T-} followed by the user determined unit name. In general, upper and lower case versions of the prefix and the name are equivalent. The entire label parameter is represented by "LTD-label". That is;

```
┌──────────────────────────────────────────────────┐
|  LTD-label  is equivalent to  {L-| T-| D-}label  |
└──────────────────────────────────────────────────┘
```

where "label" is the user-determined unit name.

An L-label refers to a LOGIC unit, a T-label to a TEST unit, and a D-label to a DISPLAY unit. Units within a file must have unique names (that is, there may not be two tests called T-lesson.one); but units may have the same name in different files (that is; a Display D-day1, a Test T-day1 and a Day L-day1 could all exist at the same time.)

There are three means by which material might be placed into a workspace:

1) The material may be copied to the workspace outside of TAIM with the MTS $COPY command.
For example: $COPY MLWT:CURRIC TO WS1

2) The material may be retrieved from the existing files or workspaces and placed in one of the workspaces with the TAIM PLACE command.

3) Material may be entered into a workspace with the TAIM EDIT and SETKEY commands.

Once a desired unit is in a workspace, it must be

labelled with the LABEL command after which it may be saved in the permanent files with a SAVE command.

In TAIM, as in the MTS EDITOR, all commands consist of a keyword followed by 0, 1, 2, or 3 parameters. Every parameter is separated from the keyword and from every other parameter by at least one blank; no blanks may appear within the keyword or within a parameter. Following are the implemented Mode Two commands.

CATALOG { L| D| T| ALL} [XREF]

A list of all available units in the Logic, Test, or Display files may be obtained by specifying the first parameter of this command as L, T, or D respectively. All three lists are produced if the first parameter is ALL. If the second parameter "XREF" is included, then for each unit listed, the name of all Logic units referencing that unit are also listed.

```
EXAMPLES:     1) catalog all
              2) cat d xref
```

DROP LTD-label

A unit which is no longer required may be removed from the TAIM permanent files through the DROP command providing it is not referenced in any LOGIC unit. LTD-label is the label (including the prefix L-, T-, or D-) of the unit to be dropped.

```
EXAMPLES:     1) drop l-lesson.one
              2) drop t-CHAPTER.3
              3) DROP D-lesson1.3a
```

<u>ED</u>IT ws#

A MODE 2 user may invoke the MTS EDITOR while running TAIM in order to alter the contents of a workspace by means of the EDIT command. Note however, that the editor can be run on any of the three workspaces outside of TAIM (that is, while in MTS) at considerably less expense. Using the EDITOR with the TAIM EDIT command should be done only when relatively few changes are to be made.

```
EXAMPLES:      1) edit ws1
               2) EDIT WS2
               3) EDIT ws3
```

While the EDITOR is an integral part of the TAIM Mode Two system, it was neither designed nor constructed by the authors of TAIM. A description of the main portions of the EDITOR may be found in Appendix D.

<u>EVAL</u>UATE ws#

Before saving a workspace in one of the permanent files, TAIM evaluates the workspace to ensure that it is acceptable. TAIM always does this evaluation in conjunction with the SAVE command; but the user may EVALUATE a workspace without having it SAVEd with this command.

```
EXAMPLE:       eval ws3
```

<u>F</u>IND LTD-label

The LOGIC file contains units which reference other LOGIC units, TEST units, and DISPLAY units. For a specific unit, it is sometimes useful to know which of the LOGIC units make reference to it. The names of all LOGIC units referencing

LTD-label are printed by the FIND command.

```
    EXAMPLES:
            1)  find  d-lesson.one        would print
    the labels  of  all  Logic  units  containing  the
    statement "SHOW D-LESSON.ONE"

            2)  f  t-ch3test         would  print the
    labels of all Logic units referencing T-CH3TEST.

            3) fi l-lesson1         would  print  the
    labels of all Logic units "leading to" LESSON1.
```

LABEL ws# { LTD-label | ? }

Every saved unit must have a label by which the MODE 2 user and the TAIM system "know" the unit. All LOGIC units have L-labels, TEST units have T-labels, and DISPLAY units have D-labels. Before any workspace can be SAVEd, it must be LABELed. The label given to a workspace becomes the name of the unit after it has been SAVEd. The prefix chosen with this command determines the TYPE (LOGIC, TEST, or DISPLAY) of the unit; and in which file it will be SAVEd. Use of the "?" for the second parameter causes the current label of the named workspace to be printed.

```
EXAMPLES:      1) l ws2 d-curric
               2) lab ws2 ?
```

NOTE that upper and lower case representations of any LTD-label are equivalent. That is; the commands:

```
    label ws3 D-CURRIC     and     label ws3 d-curric
```

will have identical results. If a user labels a workspace with a label which is currently in use in the permanent files, he is given a warning message. If he subsequently

SAVEs that workspace, no further warning is given and the workspace replaces the old version of the unit with the same name.

PLACE { NULL | ws# | LTD-label } [INTO] ws#

This command is available to allow the MODE 2 user to move units from permanent files to workspaces (for modifications); to move units between workspaces; and to clear out workspaces.

PLACE NULL INTO WS# will delete all material from the named workspace.

PLACE WS1 INTO WS2 will produce a copy of WS1 in WS2.

PLACE LTD-label INTO WS# will produce a copy of the permanent unit named LTD-label into WS#. The place command places the label on the workspace as well.

EXAMPLE: The contents of unit D-CURRIC may be altered by the following sequence:

```
pl d-curric ws1
edit ws1
: <EDITOR commands altering the DISPLAY>
:stop
save ws1
```

SAVE ws#

The SAVE command EVALUATES the named workspace, and if successful, makes a copy of the workspace in one of the permanent files. Which file is determined by the workspace label's prefix. If the unit already exists in the file, it is replaced by the contents of the workspace.

NOTE that the restrictions on the acceptability of DISPLAY units are not stringent. TEST, or LOGIC units may be given DISPLAY labels and temporarily SAVEd as DISPLAYS; they may

then subsequently be retrieved for editing or DROPped (since they would not be referenced by any LOGIC units).

SETKEY ws#

The TAIM SETKEY command is similar to the TAIM EDIT command in that its function is to place the user into a mode where further commands are accepted. When in EDIT mode, the computer types a ':' whenever it is ready to accept input; SETKEY types a '+'.

When a teacher makes a test for students, he also (but perhaps only mentally) makes a test KEY. This KEY is the standard by which the test will be scored. The only type of responses that are acceptable to TAIM-given tests are the integers one through nine. For each of the nine possible responses, the teacher has the opportunity of assigning a RESPONSE WEIGHT. For example; suppose a multiple choice test question had five possible responses (1,2,3,4, or 5), and that "3" was the correct response. The teacher could assign weights of zero to responses 1,2,4, and 5, and a response weight of one to response 3. Students would then receive a score of one on the question if their response was 3, zero otherwise. As another example, suppose a five choice item had one response (4) which was correct, and another (2) which was only partly correct. The teacher could then assign response weights of "03050" to responses 1 through 5 respectively. Students who gave response "2" would receive 3 marks (60%), those choosing "4" would receive 5 marks (100%) for that question. Students with other responses would get zero

marks for that question.

Each question has a QUESTION WEIGHT; defined as the highest possible score for that question. The question weight of the first example above is 1, of the second example, 5. The sum of all question weights on the test is called the WITHIN-TEST TOTAL. For any question, a student's percentage score is his score x100 divided by the Within-test Total.

The six commands acceptable in SETKEY mode are:

```
+CLEAR
+PRINT
+RESET
+SET
+STOP
+WEIGHT
```

• The CLEAR command sets all RESPONSE WEIGHTS, all QUESTION WEIGHTS, and the WITHIN-TEST TOTAL to zero; and the BETWEEN-TEST WEIGHT (see the WEIGHT command) to one.

• The PRINT command prints out the key in the following format:

```
                   NUMBER OF QUESTIONS    a
                   BETWEEN-TEST WEIGHT    b
         QUESTION  RESPONSE  WEIGHTS QUESTION
          NUMBER   1 2 3 4 5 6 7 8 9   WEIGHT
             1     c d e . . . . . .     i
             2     f g h . . . . . .     j
             :     : : : : : : : : :     :
             a     . . . . . . . . .     .
                   WITHIN-TEST TOTAL     k
```

Where:

a is the highest numbered question for which weights have been entered,

b is the Between-test Weight (see the WEIGHT command),

c,d,e are the response weights for responses 1,2,3 (respectively) of question 1,

f,g,h are the response weights for responses 1,2,3 (respectively) of question 2,

i̲ is the maximum number in c,d,e,... The QUESTION WEIGHT for question 1,

j̲ is the maximum number in f,g,h,... The QUESTION WEIGHT for question 2, and

k̲ is the sum of i,j,... ; the WITHIN-TEST TOTAL.

• The RESET command has two possible effects. If a KEY had been established for a particular workspace before the current issuance of the SETKEY command, then RESET acts to restore that previous KEY; replacing the current KEY. If no previous KEY exists, then RESET acts identically to the CLEAR command - setting all response weights to zero.

• The SET command causes SETKEY to print:

```
              Q#  123456789
             ?
```

The user is prompted with successive question marks to enter question numbers and response weights. Question numbers are integers 1 thru 40 and must appear directly under the "Q#"; response weights appear under the response to which they apply. Question numbers need not be in any particular order; the response weights must be blanks (read as zeros), or integers 0 - 9. When all response weights are entered, the user enters a NULL line (no characters) to leave the SET command.

```
EXAMPLE:      +set
                  RESPONSES
                Q#  123456789
              ?  1     1
              ? 3   0120
              ?  2  1   5
              ?           <null line entered>
              +print
                      NUMBER OF QUESTIONS   3
                      BETWEEN-TEST WEIGHT   1
               QUESTION  RESPONSE  WEIGHTS  QUESTION
                NUMBER   1 2 3 4 5 6 7 8 9   WEIGHT

                   1     0 0 0 1 0 0 0 0 0      1
                   2     1 0 0 0 5 0 0 0 0      5
                   3     0 1 2 0 0 0 0 0 0      2
                         WITHIN-TEST TOTAL      8
              +
```

• The STOP command (as in the EDITOR) terminates SETKEY mode.

• The WEIGHT command is used to set the BETWEEN-TEST WEIGHT for each test. Each test is given (by default) a

Between-test Weight of one. This means that each such test contributes equally to the determination of an aggregate grade when all test marks are averaged. However, for important tests, teachers may wish the students' score to contribute more heavily to an aggregate mark. This will result if the test's Between-test Weight is set to an integer higher than 1. This weight may also be set to zero so that the test score does not contribute to an aggregate mark at all. The Between-test Weight must be an integer between zero and nine (inclusive).

```
 EXAMPLE:          +we
                    ENTER BETWEEN-TEST WEIGHT
                   ?5
                   +
```

D) MODE ONE

Being the lowest in the hierarchy, Mode One would be acessible to all users registered to TAIM. Where Mode Two commands are directed towards the input, assessment, and modification of curriculum; Mode One provides facilities for the assignment of this curriculum to students and monitoring of student progress. The interface between the TAIM system and the students is controlled by the teacher entirely through Mode One.

A majority of the Mode One commands concern individual or groups of students. For these, the first parameter specifies for which student the command is to operate. For brevity, this parameter is abbreviated as "Sids" (student identifications). Whenever this parameter appears in the following syntactic descriptions, it represents the following three possible replacements.

1) Each student, when registered, is assigned a six character identification code. Commands may be made to operate on individuals or small groups of students by replacing Sids with a list of from one to five such codes separated by commas.

2) Each student, when registered, is placed in a CLASS. Each class is assigned a single CLASS-LETTER which may be any alphabet character. Commands will operate over all students in a given class if Sids is replaced by "CLASS-$" where $ is replaced by the appropriate class letter.

3) Commands will operate over all registered students if Sids is replaced by the text "ALL". In summary,

```
┌──────────────────────────────────────────────┐
|  Sids is equivalent to {list-of-ids|CLASS-$|ALL}  |
└──────────────────────────────────────────────┘
```

AVERAGE Sids [FULL]

This command allows the Mode One user to obtain the average score for a range of pupils on specific tests. When issued, it causes TAIM to prompt for the labels of desired tests; a null line terminates prompting. Output for each student in Sids is the student's name, and for each Test requested: the date the test was given, the student's score, the student's possible score, the Between-Test Weight of the test, the student's percentage score, and the label of the test. If FULL was supplied as the second parameter of the command, the label of the Logic unit from which the Test was assigned is also given. This is followed, for each student, by an average per cent score computed as the sum (over tests) of the product of the student's total score and the Between-Test Weight divided by the total possible sum times 100.

```
EXAMPLE: average macleb full
         T-LABEL : log.pwr
         T-LABEL : final
         T-LABEL :   <null line>
         STUDENT : MACLEB
         73-02-05 006/010 (1) 60.0% T-LOG.PWR L-DAY3A
         73-02-13 064/090 (2) 71.1% T-FINAL   L-DAY8
         AVERAGE  070/100     70.0%
```

DISPLAY Sids {HOLDS | MESSAGES | NOTES |REGISTRY}

This command allows the display of selected information for the range of students given by Sids. If the second parameter is HOLDS or NOTES, then a list of the current HOLDS or NOTES (see the HOLD and NOTE commands) is printed. During each run of the DRIVER, messages may be queued for the teacher. These are retrieved by issuing the DISPLAY MESSAGES command the next day. DISPLAY REGISTRY simply displays registration information (see REGISTER command).

```
EXAMPLES:     a) display all notes
              b) disp class-a m
```

HOLD Sids #   UNHOLD Sids

These two commands permit the temporary suspension of student printouts. In the HOLD command, # represents the number of class days for which Sids printouts are to be withheld. The HOLD is deleted automatically after # class days, but can be deleted prematurely with the UNHOLD command.

```
EXAMPLES:     a) hold jonesj 2
              b) unhold all
```

MARK Sids [FULL] LT-label

This command is used to retrieve student test scores. (The tests are actually "marked" by the READ command.) The "LT-label" parameter may be the label of either a Logic unit or a Test unit. For each student in Sids, a search is made for occurances of the label in his records. If found then the student's name, the date the Test was assigned, the student's score, the possible score, the student's percentage score and a label are printed. The Logic unit label is printed if the Test unit label was given in the command, and vice versa. If FULL was given as the second parameter, then for each question on the test the student's response, his score, and the possible score are also output.

```
EXAMPLES:     a) mark all t-log.pwr
              b) ma class-a full t-final
```

NOTE sids [D-label]   UNNOTE sids {# 1 EVERY}

The NOTE command permits the teacher to have specified text appear on the named student(s) next printout. If D-label is given as the second parameter, then the Display with that label appears on the printout. If no second parameter is given, then the command prompts the issuer to enter the text of the note. Between the time the Note is queued for the students and the run of the DRIVER, the Note may be deleted with the UNNOTE command. The DISPLAY NOTES command printout includes the number (based on order of issuance) of each Note. This number must be given as the second parameter of UNNOTE to delete a specific Note; EVERY deletes all Notes

queued for Sids.

```
EXAMPLES:     a) note smithb d-explanation
              b) note all
                 ENTER NOTE: there will be a quiz next day

              c) unn class-a 2
```

<u>POINT</u> Sids L-label

There are two means by which students may proceed from one Logic unit to another. One is by following the GOTO statements in each Logic unit (this may be considered as a default procedure); the other is by having a Mode One user explicitly direct or point the students to a Logic unit. This command causes the named student(s) to have their next-logic unit pointer (or next-lesson pointer) set to L-label.

```
EXAMPLES:     a) point all L-start
              b) point class-a L-mid-term
              c) point smithb l-lesson.4
```

<u>READ</u>

This command activates a procedure which reads-in student cards. For each card read, the student is identified and his last "record" located. If he had been given a test, then the test key is retrieved and his responses marked; the results are placed in his record file. Then his last Logic unit is referenced, the decision algorithm portion executed to determine his next-lesson pointer, and this pointer placed in his last record.

REGISTER s:code   UNREGISTER s:code

The REGISTER command activates an interactive routine for the registration of students to the TAIM system. The parameter on this command is the six character student identification code to be assigned to the student. If the "s:code" code is already registered to the system then a similar routine allowing modification of registration is invoked. UNREGISTER simply deletes a student's registration.

```
EXAMPLE:      reg jonesb
                 NAME: brian r jones
                  AGE: 17
                  SEX: m
                CLASS: a
              COMMENT: average student
```

TRACE Sids [FULL] [#]

This command is provided to trace the history of TAIM assignments to students. If only the first parameter is given then for each student in Sids, the label of the last Logic unit and the date this unit was entered are printed. If FULL is specified then the Notes, Tests and Displays administered during that unit are also listed. If the "#" parameter is given, then the trace begins that number of class days back in the record files.

```
EXAMPLES:     a) trace all full
              b) tr class-a 3
              c) tr jonesj full 99
```

<u>WHO</u> Sids [<u>MISS</u>ED] LTD-label

This command allows the Mode One user to determine which of the students in Sids was given (or was not given) a specific logic, test, or Display unit. If MISSED is omitted then the identification codes of every student receiving the unit with the given label are printed. If MISSED is included, then the IDs of all students who did not receive the unit are printed.

3. <u>THE OFFLINE INTERPRETIVE DRIVER</u>

While the MONITOR program discussed in the previous section has a great many diverse types of actions, the duties of the DRIVER are relatively simple; its main task is to print (on a high speed printer) lessons for pupils. The information necessary to assemble these printouts is available in the systems files.

A) <u>DRIVER OPERATION</u>

The STUDENT file contains an index consisting of each student's registration data. The index is arranged by student identification code in alphabetic order. The DRIVER proceeds through the index, performing all necessary tasks for each student before moving on to the next, until the list is exhausted.

Obtaining the next index record, the DRIVER first references the HOLD file to see if this individual, his class, or ALL students are to be held. If a HOLD is in effect the DRIVER produces a warning message and goes on to

the next student.

If the student is not held, the DRIVER continues. Each index record contains an entry which enables the DRIVER to locate the student's last record. This record should contain a next-lesson pointer. (If it does not, the program chains back through previous records to locate the most recent pointer.) The DRIVER then queues lines which will print the student's identification code, his full name, and his pointer on a new page. If this last record indicates that one or more tests were given, the results of these are queued for printing for the student. The next-lessons pointer is actually the label of a unit in the Logic file. The DRIVER opens a new record for the student in the STUDENT file and begins interpreting the referenced Logic statements. Each statement causes an entry in the student's record. If the statement is a SHOW, the appropriate DISPLAY is queued for the student; if a TEST, the Test text from the TEST file is queued; if a MESG, the message text is queued in the MESSAGE file (the student does not see this message); if an IF, the most recent information is retrieved from the STUDENT file and the condition evaluated; if a WHEN or GOTO, then this signals the end of processing for this student.

In producing the student printouts, five possible error conditions may arise.

1. Evaluation of a condition on an IF statement may not be possible because the student had not taken a referenced test. The condition is evaluated as if the student received zero on the test.

2. A SHOW D-label statement may be encountered where the D-label did not exist.

3. A TEST T-label statement may be encountered where the T-label did not exist. Errors 2 and 3 cannot occur unless a Mode Three user has improperly rearranged the files. The Mode Two portion of the MONITOR will not permit either the saving of a Logic unit which references non-existent Displays or tests nor the deletion of displays or tests referenced by Logic units.

4. A NOTE command may specify a Display which did not exist. This error could arise only if the needed Display was deleted between the time the NOTE was issued and the next run of the DRIVER program.

5. Evaluation of an IF condition may not be possible because although the student had been given the test, it was never marked. The condition is evaluated as if the student received zero on the test.

If any of the above errors occur, all relevant information is queued so that the mode two or Mode Three users of TAIM can take corrective action. Along with these possible errors are three possible warnings that the DRIVER may produce.

1. A warning message is produced if, for some reason (usually because the student was absent), a student's next-lesson pointer had not been set. He would be assigned the same lesson as last day. (The next-lesson pointer is set by either reading in the student's response card, or POINTing the student.)

2. A warning message is produced for each student whose printout was suspended by a HOLD command.

3. A warning message is produced whenever a HOLD command expires, warning the teacher that the student will be released.

After completing processing for all students, the DRIVER has accumulated the student printouts on one file, and program errors, messages and warnings on a second. During processing it also accumulates statistical information on its actions on a third file. As its final

tasks, the DRIVER transfers statistical records generated by the MONITOR to the statistics file and places any MONITOR commands queued for offline processing into a fourth file.

The generated statistics are then transferred to computer tape for later analysis. The errors, messages and warnings are sorted into a meaningful order and printed. Each line of the student lessons is prefixed by the student's class letter, identification code and a record number. These are sorted alphabetically within classes, the sorting information removed, and printed. The commands queued for the MONITOR are then arranged in order of user, and the MONITOR initiated to process these commands.

B) LESSON PRODUCTION

The actual production of each printout or lesson for each student is done through retrieving, interpreting, and executing the appropriate statements in the Logic file. SHOW and TEST statements cause the relevant Display or the textual portion of the relevant Test to be printed for the particular students. These are the only two Logic statements that cause text to be produced, but additional text may be queued for inclusion in students' printouts with the Mode One NOTE command. Each student's lesson is produced independently of other students registered to TAIM. Processing of the Logic statements ends for each student when a WHEN or a GOTO statement is encountered in the current Logic unit.

4. COMMENTS ON THE DESIGNED SYSTEM

The first three sections of this chapter have presented a description of the TAIM system as it was implemented. The purpose of this section is to provide a justification for the chosen MONITOR commands to explore the relationship between the designed system and related CMI systems.

A) ON THE MONITOR COMMANDS

The three modes in the MONITOR can be considered as independent of each other. The purpose of Mode Two is to facilitate the construction of curriculum; the purpose of Mode One is to enable a classroom teacher to manage the instruction available through TAIM for one or more classes.

Consider a Mode One teacher and a class in an environment where individualized instruction packets, criterion tests, and rules for progress, and sequencing are well defined. (An environment such as this exists with the IPI system, see Oliva, Maguire, & Pacey, 1973; and the EMDA system, see EMDA Committee, 1971.) The role of the teacher here includes the marking of tests and the assignment of packets according to the progress and sequencing rules. These tasks are done by TAIM automatically.

Any such system cannot always perform perfectly. That is, there will be instances when the instruction being provided for students is not appropriate. This can occur in three ways.

In the most severe breakdown of the system, suppose

that, for a particular student the existing program does not contain any appropriate instruction. Here the teacher has no choice but to remove the student from the program, and provide personal help until the student can re-enter. In TAIM, this temporary removal of the student is possible through the HOLD command.

At a less serious level, appropriate instruction may be available in the program but the student is just not presently at that part of the course. This may be remedied by the teacher simply deciding to assign the appropriate instruction. Using TAIM, the teacher accomplishes this with the POINT command.

Finally, the student may be receiving appropriate instruction, but could also use remedial or enrichment work available in the program. The teacher can make these supplementary assignments; under TAIM this would be done with the NOTE command. If supplementary material is not available, the teacher must devise and assign these manually under any system.

As students proceed through an instructional system such as this, they generate two types of information. One is the path they take, which lessons and tests they are given; the other is the result of the tests they receive.

Student histories are usually kept under all individualized systems. TAIM allows two methods by which this information can be retrieved. One sort of information which the teacher might require is a history (at some level

of detail) of a particular student. Under most systems this type of data is kept; it may be retrieved by a TAIM user with the TRACE command. The other sort of possible query is; for a particular lesscn, instruction packet, or test; who did or did not receive it. This data may be readily retrieved with the TAIM WHO command.

Concerning students' test marks, three possible types of information appear to be necessary. At the simplest level, a teacher may want to know an individual student's mark on a particular test. The MARK command is provided for this (It may also be used to generate all students marks on a particular test.) At a second level, the teacher may want to compute a weighted average mark over a number of tests for an individual pupil. The AVERAGE command accomplishes this task. At a third level, a teacher may want average marks computed over a number of tests and/or over a number of students. Because of an underlying philosophy of individualization of instruction, TAIM does not provide this facility (however it would be relatively easy to implement).

Consider now, a <u>Mode</u> <u>Two</u> teacher, or perhaps a curriculum consultant who has the task of either creating new or repairing existing paths, packets, and tests in a course. At a global level, he may require a list of all available packets and tests, and particularly how they are currently joined together to make up the present course. Under TAIM, an up-to-the-minute catalog of all available units with a cross-reference index is generated by the

CATALOG command. At a more local level, he may want to know the point in a course where a certain logic point, curriculum packet or test is referenced. The TAIM FIND command retrieves this information.

A teacher altering a course, especially the logic of a course, must be concerned with the problems of deleting required portions of a course, inserting references to materials that do not exist, or providing incomplete information. As previously explained, TAIM will not permit a majority of these oversights.

Very few Mode Three commands were implemented because the actions required were simple to perform through the resident MTS system. For example, by invoking the file editor on the TAIM SYSTEM file: any error message could be modified or replaced; any command name, abbreviation, status (active/inactive) or number could be altered; the I/O assignments for the MONITOR or DRIVER could be changed; and the default initialization for any file reset.

Two Mode Three commands envisioned but not implemented were DUMP and CONDENSE. The DUMP command was to cause an offline dump of a particular file. These dumps were needed for the Display and Test files; time pressures forced production via an independent utility program rather than a command verb. When a great deal of adding and deleting of units is done in any file, unreferenced "garbage" records are left behind. The CONDENSE verb would have caused an optimum restructuring (and reordering) of files. In the

implemented version, this task was done manually.

B) ON RELATED CMI SYSTEMS

Three current CMI systems were discussed in Chapter Two: PLAN (Program for Learning in Accordance with Needs), IMS (Instructional Management System), and the CONWELL Approach. The purpose of this section is to compare these systems to TAIM.

At the most general level, there appear to be two advantages of the TAIM system over other presently operating CMI systems. The first, and perhaps most important, is that under TAIM, the program or course logic is readily alterable. Because of this flexibility, it appears to the writer that instructional logic could be written to make TAIM behave as if it were PLAN or IMS or the CONWELL System.

The second main advantage of the TAIM system is that it provides more control over course content for the teachers using the system. The other CMI systems reviewed provide teachers with all of the power provided by Mode One of TAIM, but reserve the capabilities of Mode Two for curriculum experts. Arguments presented in section one of Chapter Two show why this control should be delegated to teachers using the system.

Unfortunately, flexibility of logic is a double-edged sword. The extreme flexibility provided by TAIM makes it necessary to impose some external order to course sequencing. The interrelatedness of even twenty Logic units under TAIM may be impossible to comprehend if presented

without an organizing context. Anyone using TAIM may be forced to adopt a context such as that used in the CONWELL Approach. An alternate context is discussed in Chapter Six.

C) ON THE ADVANTAGES OF TAIM

TAIM is a system for augmenting the effectiveness of the classroom teacher. The advantages of TAIM accrue to the teacher through improvements in his power to control curriculum and to manage instruction. Advantages of TAIM to students and to the remainder of the school system result indirectly from these advantages to the teacher.

In the construction of curriculum, TAIM facilitates both the accumulation and the iterative improvement of instructional sequences and materials. In the current traditional system, each teacher prepares a number of lessons for a class during the school year. While a few of the materials may be saved, the major part of each lesson is not available to other teachers in the school system.

Using TAIM, the major part of each lesson is saved and available to other teachers. This availability not only obviates the need to re-generate the lessons every year by every other teacher, but defines a stable entity which can be evaluated and improved. The "entity" nature of the TAIM-held lessons permits the examination of each lesson in and of itself, and also the relationships existing between it and associated lessons. This property may be essential for the construction of complex individualized curriculum (see Simon, 1962, pp. 467-482), yet it is not clearly present in

any other instructional systems known to this writer.

The execution of well-planned individualized instruction requires a considerable amount of bookkeeping. Those who construct the curriculum devise rules such as: "Whenever a student reaches this point, he will take either path 'A' or path 'B' depending on factor 'X'." When information on factor 'X' becomes available, it must be saved; and when any student reaches the specified point, both the rule and the factor data must be retrieved to make the decision.

The monotonous task of storing and retrieving the data and rules is comparable to the dehumanizing tasks of an assembly line worker. Under TAIM, the entire procedure is done automatically by computer - the teacher must focus on his unique talents: providing students with the human contact that cannot be simulated and providing the supervision and continuous monitoring that the computer needs to remain adept at its job.

> ... if the organization expects its members to be committed, flexible, and in good communication with one another ... it cannot merely pay more money, ... there must be the possiblity of non-economic rewards such as autonomy, genuine responsibility, opportunities for challenge and for psychological growth (Schien, 1970, p. 127).

The TAIM Mode Two facility provides the challenge and the opportunities for responsibility and growth of teachers, limited only by their ingenuity and desire.

CHAPTER FOUR

PROCEDURES FOR THE STUDY

The purpose of this chapter is to outline the procedures that were used for an initial feasibility testing of the TAIM system. Of four sections following, the first provides a rationale and delimits the procedures; the remainder discuss the two procedures and cost accounting.

1. RATIONALE AND DELIMITATION OF PROCEDURES

The procedures used in evaluating the TAIM system were circumscribed by two external factors: cost and duration. Since very little money was available for testing the system a minimum equipment configuration was chosen and the student sample was restricted to about fifty. The overall duration of the evaluation was restricted to six months.

In the evaluation of an instructional system such as TAIM, two general questions are of paramount importance: "How effective is the instruction?", and "How much does it cost?" A system which does not provide adequate instruction is unnacceptable regardless of cost; but even an extreme improvement may not be feasible if the cost is prohibitive.

Perhaps the best test of the effectiveness of

instruction is an assessment of its outcome; i.e. a measure of how much the students learn. Instructional outcome (or learning resulting directly from teaching) is dependent on a number of factors; the most important of which is the content of that instruction.

This type of assessment was not chosen because a) it would not be reasonable to expect newly-developed instructional materials to provide optimum instruction, and b) an assessment of this kind would be an evaluation of the constructed curriculum materials directly and of the TAIM management system only indirectly. This is not to contradict the hypothesis that better instructional materials (and therefore better instruction) may be provided using TAIM. However an assessment of this phenomenon would require curriculum work over an extended period of time (say three years) before meaningful results could be obtained. Similarly, a lengthy interval (perhaps one year) would be necessary to establish and evaluate comparitive attitudes toward the system by parents, teachers, and students.

However, it is reasonable to expect that instruction will not deteriorate as a result of improved instruction-management facilities. The above arguments led to the formulation of the following guiding question.

> Would the system work as it was intended to work? That is, would the computer, the teachers, and the students proceed in the described ways and accomplish their respective tasks in an efficient and enjoyable manner?

The second general question, concerning cost, was also

difficult to answer within the limitations of the evaluation. Costs can be divided into four general categories: hardware, computing, staff, and school resources. While the hardware requirements can be estimated and quotations requested to produce fairly accurate cost estimates, the remaining three categories of cost depend heavily upon how the system is used. The following quiding question was chosen.

> What was the cost of the system as constructed and as used in each of the four categories during implementation and evaluation?

An adequate feasibility evaluation of the TAIM System is seen as a more protracted procedure than that attempted here. However the two guiding questions above would continue to be the focus of further assessment. The procedures used in this study constitute merely the first step of an iterative process. They may show that TAIM is not feasible; but more likely they will provide formative information enabling improvements to the system so that further feasibility assessments would be indicated. Only after several of these small studies had completed successfully would a full scale implementation be desirable.

Two procedures were used. The first involved the training of six Education students in the use of TAIM and having them design and input a set of curriculum materials to the system. This is called the Preparation procedure. The second, the Implementation procedure, consisted of using TAIM and the curriculum prepared in an actual teaching

situation. Cost accounting information was collected during both of these. Following is a list of the dates of events central to the evaluation procedures:

September (1972): Mr. Robert Motiuk, a teacher at St. Joseph's Composite High School (E.C.S.B.) was contacted. Motiuk and the writer selected a section on logarithms from a Math 20 course for implementation testing. A flowchart of the content and sequence of eight days of instruction was devised (see Appendix C). Background and Attitude Questionnaire administered to Motiuk and Mr. Mel Fischer.

October 26: Began training members of Ed. C.I. 466 in the use of TAIM. Administered Background and Attitude Questionnaire.

November: Ed. C.I. class constructed and input curriculum materials as outlined by the flowchart

December 13: Post-Attitude questionnaire and TAIM Mastery test administered to Ed. C.I. 466 class. End of Preparation procedure.

January 10-19 (1973): Instruction managed by TAIM for logarithms section of Math 20 class. Student attitude tests administered.

January 22-31: Materials modified by teachers and the writer for use in a Math 33 class.

February 6-15: Instruction managed by TAIM for logarithms section of Math 33 class. Student attitude tests administered.

February 16: Post-Attitude test administered to Motiuk and Fischer.

February 19: Recorded interview with Motiuk, Fischer, Dr. T.E. Kieren and the writer.

2. PREPARATION OF CURRICULUM

This first procedure was planned essentially to field test the construction of curriculum materials and the use of the Mode Two portion of the MONITOR. The procedure consisted of two parts: training a group of teachers to use the TAIM system, and then having them design and input curriculum materials.

The group of teachers were actually teachers-in-training; members of a University senior education course (Ed. C.I. 466: The Use of Computers in the Teaching and learning of Mathematics) under the direction of Dr. T. Kieren. The six class members were: S. Baird, D. Schmermund, A. Liu, E. Williams, K. Palylyk, and B. Derman. The one-semester course lasted thirteen weeks: the first seven taken by Kieren for teaching introductory computer-applications principles, laboratory applications specifically, and the BASIC programming language (see Hatfield, 1968); and the latter six weeks taken by the writer for teaching the use of TAIM.

The class met for an hour and one half twice each week. At the first meeting of the TAIM portion, a Background and Attitude questionnaire was administered. At each of the following four meetings, a lecture and a computer produced printout were given. The topics of the lectures (in sequence) were: "An Overview of the TAIM System," "A Rationale of the TAIM System," "Curriculum Construction and the Use of Flowcharts," and "The Use of Files under TAIM."

Each computer printout contained material describing portions of TAIM, and directions for a laboratory activity. The topics of the printouts (in sequence) were: "The MTS System," "The MTS File Editor" (printout reproduced in Appendix D), "Carriage Control and Special Characters in Computer Printing," "The Mode Two System," and "The Mode One System."

At the next meeting, the curriculum flowchart and Math 20 textbooks were distributed. The remainder of the meetings were devoted to planning the details of the eight days of instruction, assigning tasks, and discussions of implementation procedures. The members of the class were asked to keep time-sheets of their activities, and to prepare a criticism of TAIM Mode Two. At the final meeting, the Attitude portion of the Background and Attitude questionnaire was re-administered along with a formal examination on members' knowledge of TAIM Mode two. The Preparation procedure lasted for about eight weeks and was completed in early December of 1972. Results are reported in the next chapter.

3. CLASSROOM IMPLEMENTATION

The second procedure consisted of using TAIM for the management of instruction using the curriculum prepared by the Ed. C.I. 466 class. The system was implemented first for eight school days with a Math 20 class under Mr. Motiuk and then for a further seven days with two Math 33 classes under

Mr. Fischer at St. Joseph's Composite High School (Edmonton, Alberta). The original curriculum specification flowchart was prepared by Motiuk and the writer (see Appendix C). The materials prepared by the Ed. C.I. 466 class were reviewed by Motiuk just prior to use with his class; he requested several minor alterations which were made by the writer.

A Texas Instruments Silent 700 portable terminal and an Omnitex accoustical coupler (data set) from the Division of Educational Research Services (University of Alberta) were placed in the school. The procedures for connecting the terminal via a telephone line to the University of Alberta's IBM /360-67 and invoking the TAIM MONITOR were explained; and Motiuk registered his class on the system. Motiuk and Fischer were given the same Background and Attitude questionnaire as was given to the Ed. C.I. 466 class.

On the first class day, a pre-achievement test on logarithms was given to the students. For the remaining seven class days students received TAIM printouts and returned response cards which were typed into the MONITOR on the terminal by the teacher. Motiuk learned the use of the TAIM Mode one subsystem through the experience of using it under supervision by the writer. On the last class day, a post-achievement test on logarithms was given; following this, a student reaction questionnaire was administered and a discussion period conducted by the writer with the students.

While students were receiving their lessons, Motiuk and

the writer noted weaknesses in the curriculum. Afterwards Motiuk, Fischer and the writer reviewed the entire set of materials and decided on general improvements and adaptation for the Math 33 class. (Math 20 is intended for academic-route grade eleven students; Math 33 is provided for technical-route grade twelve students.) A number of changes were made, the majority to the curriculum logic.

The system was then implemented with two Math 33 classes simultaneously for seven school days. For both of these classes, a pre- and post-achievement test, and student attitude questionnaire were given; and after final testing, a discussion was conducted with the students by the writer. At the end of this procedure, a taped interview was held with Motiuk, Fischer, and Kieren. Following this the Attitude portion of the Background and Attitude questionnaire was re-administered to Motiuk and Fischer.

4. COST ACCOUNTING

One of the guiding questions of this study concerned the cost of using the system. Costs of running TAIM arise in four separate areas: hardware, computing, staff, and school resources.

In the area of hardware, costs were calculated on both the equipment actually used and on what might be considered ideal equipment.

Data on computing costs were collected in a number of ways. Both the TAIM MONITOR and DRIVER were constructed to

produce statistical records which included cost information. These records enabled a frequency count of the issuance of MONITOR commands and calculation of their average cost. Similar calculations were made from the DRIVER statistics. As well, each offline run of the DRIVER produced a printout of detailed actual charges. These records enabled a fairly accurate representation of the cost of using the prototype system in the school. It must be kept in mind however, that these costs would vary widely depending on the host computer, and the language used and effort expended in optimizing both the MONITOR and DRIVER programs. Costs are based on the actual charges levied by the University of Alberta Department of Computing Services (see Appendix E).

Staff (people) resources are needed during both the curriculum preparation and in-school implementation of the system. An initial system expense would be incurred at the outset to train both Mode One and Mode Two users (teachers). Then considerable time would be necessary for the Mode Two users to design, construct, and input the curriculum content. Finally, there is the staff required to actually run the system for the students.

Data for the estimation of these costs were collected during the preparation and implementation procedures. The Ed. C.I. 466 teachers were asked to keep records including both the time they spent learning about TAIM and the time they spent doing curriculum work. They are also asked to suggest the format and time required for a training program

to enable teachers to accomplish these tasks. Similar time estimates were obtained from the two teachers involved in the implementation procedure.

The school resources used during implementation were minimal. Over the resources required for the regular operation of a class, only the part-time use of an office where the terminal was kept was required. School resource costs might have been a larger factor if the curriculum had been designed to make more use of external facilities; but since this was not the case, this category of cost was dropped from the accounting.

CHAPTER FIVE

RESULTS

The purpose of this chapter is to present the data resulting from the procedures outlined in the previous chapter. Comments on and analysis of these results are collected in Chapter Six. The presentation is organized with all results obtained from the Preparation procedure in section one, results from the Implementation procedure in section two, and the costing information across both procedures in section three.

1. PREPARATION RESULTS

As explained in Chapter Four, there were four sources of information concerning the Preparation procedures: a Background and Attitude questionnaire (the Attitude portion given both pre and post); time charts; TAIM Principles exam results; and written Critiques prepared by the Ed. C.I. 466 participants. These are discussed in that order in the following four parts of this section. There were six people in the class; these are herein identified Pa, Pb, Pc, Pd, Pe, and Pf.

A) THE QUESTIONNAIRE

The first ten questions of the Background and Attitude Questionnaire requested background information; the remaining twenty-five attempted to sample the teachers' attitudes. The actual Background Questionnaire questions and the responses given by the teachers are included in Appendix C. From this questionnaire, it was learned that four of the six participants of the Ed. C.I. 466 class had previously taken courses which involved the use of computers, and two had used an online file editor. Four of the teachers were completing the final year of their B.Ed. program, and two were doing post-graduate work (one with the Department of Mathematics). Only two (Pa and Pf) had teaching experience beyond student teaching.

Questions 11 through 35 comprised the attitude portion of the questionnaire. The purpose of giving this questionnaire was to sample, in a general way, the participants' attitude towards systematized instruction and technology. This instrument was used as both a pre-test and a post-test.

The twenty-five questions were designed in three categories to obtain information about three distinct but related questions. Each question consisted of a statement to which the Ps answered: "1" for strongly disagree, "2" for disagree, "3" for agree, and "4" for strongly agree. For questions worded in such a way as to support the category concept, these responses were taken as their scores. For

questions worded in opposition to the category concept, a score obtained by subtracting the response value from five was used; so that a score of 1 indicated antithetical feeling, and a score of 4 indicated support of the category concept.

It seems likely that if a teacher was "afraid" of machines, or felt that learning about computers was (*a priori*) too difficult, that this would be detrimental to learning about TAIM. It also might be expected that if such an entering bias was present, it would be reduced through the experience of learning about a computer system. This general question was the subject of the first category which might be typified by the question: "Does the participant feel personally 'threatened' by computers?"

The second category of questions attempted to assess whether the teachers saw more widespread use of computers for instruction as desirable. Effective use of computers may enable the accumulation of data and the distribution of more appropriate instruction based on this data; but it also implies a loss of privacy for the students and may be seen as a threat to personal communications and "human values." While an individual may "like" computers (category one), he may at the same time disagree with their applications to instruction. The experience of using TAIM may cause some reassessment of this attitude. The concept of this category is typified by the question: "Does the participant feel that greater use of computers should be made for instruction?"

While the two categories above are closely related, the third category of questions included in the attitude portion of the questionnaire attempted to assess teachers' perception of the appropriate means for the individualization of instruction. Some people see individualization resting on careful pre-planning and extensive data, others see it as a laisez-faire where students mostly follow their personal inclinations. A teacher favoring the latter may initially feel antithetical to TAIM; however, again, using TAIM may cause a change in this attitude. The concept of this category is typefied by: "Does the participant feel that more pre-planning of instruction and more accumulation of data is desirable?"

The three categories contained 8, 8, and 9 questions respectively. Within each category, half the statements were worded positively, half negatively. These were arranged in random order and numbered from 11 through 35 on the questionnaire. The actual question numbers are maintained in the following presentations; positively worded statements are given first within each category. To the right of each question number are two sets of four digits corresponding to results on the pre and post tests. In each set, the first digit is the number of teachers with score 1 on the question; the second digit, score 2; and so on so that each set adds to six. In general, numbers in the first two positions count those teachers with feelings antithetical to TAIM (for example, dislike of computers); numbers in the

last two count the number of teachers with attitudes favorable to TAIM (for example, support of data accumulation).

Category one, "Does the participant feel personally 'threatened' by computers," questions and scores were as follows.

PRE	POST	
0132	0033	18. If students can learn faster and better with computers, then we should use computers for instruction.
0042	0051	28. Working with computers is fun.
0510	0330	29. A computer could do a majority of teaching tasks much more quickly and reliably than a teacher.
0033	0033	35. I hope that computers will be used more in education.
0312	0132	17. Using computers makes me nervous.
1311	0213	19. I might foul-up a computer and then not be able to fix it.
0060	0051	26. Students taught by computers are more nervous.
0033	0042	33. Being taught by a computer is degrading.

On the pre-test, half or more of the teachers received scores of 1 or 2 on three of the eight questions (29, 17, and 19). On all of these there was a shift to scores of 3 and 4 on the post-test. Only on question 29 did half of the teachers score 1 or 2 on the post-test. This seems to indicate that a) the teachers did initially feel slightly threatened by computers, b) this threat was diminished after using TAIM, and c) while not personally threatened, half of the teachers felt professionally threatened even after using

TAIM.

Category two, "Does the participant feel that greater use of computers should be made for instruction?", questions and scores were as follows.

PRE	POST	
0420	1410	16. It may be possible to determine all the related variables to enable prediction of the "best" instruction for a student.
0330	0411	22. A computer can react differently to each of thirty pupils, but a teacher usually can't.
1140	0231	24. Teachers need recorded data to determine individual differences.
0141	0231	34. More detailed records of students' learning history must be kept.
0150	0150	13. Our educational system is presently operating adequately.
1410	1500	14. Teachers learn to "sense" what each child needs.
0051	0231	23. The use of computers in education will lead to a decreasing emphasis on human values.
0240	0330	27. Most teachers keep adequate records on students.

In this category of questions, the pre-test results indicate that the teachers tested were, as a group, fairly cautious about more widespread use of computers. Comparison of the post-test results indicates that after using TAIM the teachers tended to be more against the use of computers for instruction.

Category three, "Does the participant feel that more pre-planning of instruction and more accumulation of data are desirable?" questions and scores were as follows.

PRE	POST	
0150	0231	20. Few teachers plan their lessons thoroughly enough.
0042	0051	25. Systematic and thorough pre-planning for instruction is necessary.
0132	0321	30. Before presenting a concept, teachers must insure that every student has mastered all of the supporting concepts.
0231	1140	32. Detailed lesson planning is necessary to good instruction.
0213	0231	11. A computer can't really praise or encourage a child.
0411	0321	12. Many educational activities can't be planned in advance.
0330	1230	15. Behavioral objectives limit the scope of instruction.
0033	0051	21. Using computers makes instruction unnecessarily complicated.
0240	0330	31. Group activities are more valuable than individualized instruction.

While the pre-test results show a generally favorable attitude to the preplanning of instruction and accumulation of data (39 positive responses of a possible 48), after using TAIM, this attitude deteriorated slightly (to 36 of 48). Table 1 presents another view of this attitude questionnaire data, giving the counts of scores for each teacher in the three categories.

TABLE 1

PREPARATION ATTITUDE QUESTIONNAIRE SCORES

Ps		CATEGORY ONE SCORES				CATEGORY TWO SCORES				CATEGORY THREE SCORES			
		1	2	3	4	1	2	3	4	1	2	3	4
Pa	pre	0	1	7	0	0	3	5	0	0	2	7	0
	post	0	1	6	1	0	5	3	0	0	3	6	0
Pb	pre	0	2	5	1	0	1	6	1	0	5	4	0
	post	0	1	4	3	0	4	3	1	0	3	6	0
Pc	pre	0	4	4	0	1	2	5	0	0	3	5	1
	post	0	1	5	2	1	4	3	0	0	3	5	1
Pd	pre	1	1	3	3	0	2	6	0	0	0	4	5
	post	0	0	5	3	0	3	4	1	0	1	5	3
Pe	pre	0	3	2	3	0	5	2	1	0	1	4	4
	post	0	3	5	0	0	6	2	0	0	3	6	0
Pf	pre	0	1	1	6	1	3	4	0	0	4	3	2
	post	0	0	2	6	1	4	2	1	2	3	2	2
TOTALS													
	pre	1	12	22	13	2	16	28	2	0	15	27	12
	post	0	6	27	15	2	26	17	3	2	16	30	6

B) <u>TIME CHARTS</u>

At the beginning of the Ed. C.I. 466 class, the teachers were given a chart on which they were asked to keep a record of their time spent. The chart contained a line for each day from the start to the end of the class and four columns in which the teachers could enter hours spent. They were asked to distinguish whether they spent their time

working at a terminal or not; and in either case whether the work was directed towards learning about TAIM or towards the preparation of instructional materials. The time accounted was in addition to the two class meetings per week. Table 2 shows one break-dcwn of the total time spent by all users.

TABLE 2

TIME SPENT BY MODE AND PURPOSE (HOURS)

	MODE		
PURPOSE	Terminal	Non-terminal	TOTAL
Learning TAIM	10.5	20.3	30.8
Preparing Materials	93.2	67.6	160.8
Total	103.7	87.9	191.6

The duration of the class was divided into two equal intervals to emphasize the redistributicn of time spent by teachers as they became familiar with TAIM. As might be expected, during the first interval a larger proportion of their time was spent learning TAIM than during the second interval. This is depicted in Table 3. A similar shift frcm non-terminal work to work at the terminal is shown in Table 4.

TABLE 3

TIME SPENT BY PURPOSE AND INTERVAL (HOURS)

INTERVAL	LEARNING TAIM	PREPARING MATERIALS	TOTAL
First half	16.8	68.9	85.7
Second half	14.0	91.9	105.9
Total	30.8	160.8	191.6

TABLE 4

TIME SPENT BY MODE AND INTERVAL (HOURS)

INTERVAL	TERMINAL	NON-TERMINAL	TOTAL
First half	35.1	50.6	85.7
Second half	68.6	37.3	105.9
Total	103.7	87.9	191.6

TABLE 5

AVERAGE TIME SPENT BY TEACHERS (HOURS)

	LEARNING TAIM		PREPARING MATERIALS		
Ps	Terminal	Non-terminal	Terminal	Non-terminal	TOTAL
Pa	2.0	8.5	13.0	14.0	37.5
Pb	2.4	4.9	13.4	8.2	28.9
Pc	2.0	0.0	21.9	15.6	39.5
Pd	1.5	3.5	21.3	10.5	36.8
Pe	2.0	3.0	15.0	15.0	35.0
Pf	0.6	0.4	8.6	4.3	13.9
Total	10.5	20.3	93.2	67.6	191.6
Average	1.8	3.4	15.5	11.3	31.9

C) TAIM PRINCIPLES EXAMINATION

At the last meeting of the Ed. C.I. 466 class, a formal examination was given to assess the degree to which the teachers had mastered the use of the TAIM system. The exam consisted of twenty questions; each with a weight of five marks. The questions were grouped in four categories; five questions to each category. The question numbers are given below with the category descriptions, and the entire test is reproduced in Appendix C.

Questions in the SYNTAX category (7,8,10,14,15) required teachers to be able to recognize defined terms and

be able to construct acceptable TAIM commands. SEMANTICS category questions (1,2,3,9,13) required answers showing that the teacher understood the function of single or groups of commands. SYSTEM category questions (4,5,6,16,17) enabled teachers to demonstrate knowledge of the context of commands in the overall operation of the system. Questions classed as PERIPHERAL (11,12,18,19,20) were those which assessed teachers' knowledge of concepts that were necessary to the operation of TAIM but were not a part of the TAIM system.

TABLE 6

TAIM PRINCIPLES EXAM RESULTS

Ps	SYNTAX	SEMANTICS	SYSTEM	PERIPHERAL	TOTAL
Pa	17	15	17	17	66
Pb	22	24	13	24	83
Pc	11	12	22	20	65
Pd	19	23	17	19	78
Pe	23	25	25	23	96
Pf	19	22	18	22	81
Total	111	121	112	125	469
Average	19	20	19	21	78

D) CRITIQUES

At the last class meeting, the paricipants of Ed. C.I. 466 were asked to submit a critique of the TAIM system. A handout suggesting a format for the critique was supplied and is reproduced in Appendix C. The teachers were asked to comment on the two main topics of the class: that part of the TAIM system they had learned, and the process of designing and constructing instructional materials.

LEARNING TAIM

The teachers learned the Mode Two portion of TAIM. This mode facilitates twelve commands as discussed in Chapter Four. The teachers expressed the following criticisms of the commands.

1. The implemented CATALOG command printed a list of all units in the Logic, Display or Test files (with a cross-reference to the Logic file if requested). Concern was expressed that as these lists became longer, some means of requesting partial lists would be desirable. Teachers used this command to "see what was in stock" and to affirm that actions requested by SAVE and DROP commands had been executed.

2. Most teachers felt that after issuing a DROP command, the system should request a confirmation before actually deleting the unit.

3. Teachers were discouraged from using the EDIT facilities available in TAIM because the same work could be done with the MTS editor at considerably less expense.

However they did use it to "check if the right material was in the right workspace."

4. Because of the small total number of units, teachers could obtain the information produced by the FIND command with a CATALOG cross reference printout. However, the necessity of FIND when the number of units became large was recognized.

5. Some teachers felt that the LABEL command should force more structure on the names given to units. They appreciated the commands ability to warn of duplicate use of labels and to retrieve workspace labels with the question mark parameter.

6. One teacher felt that the PLACE command should have the power to access files external to TAIM and that possible concatenation of subject files would be desirable.

7. Teachers generally commented favorably on the checks that the SAVE command makes before transfering a workspace to a permanent file.

8. While teachers were pleased with the SETKEY command generally; one felt that the possibility of negative response weight should be permitted, and several felt that responses of a,b,c, etc. should be used rather than the present 1,2,3,... .

9. All teachers used the WHY command frequently. Some felt the responses produced were sufficient, but others saw need for improvement.

10. Two teachers commented on the names of commands.

They felt that for example the QUIT verb should have been STOP to correspond with STOP in EDIT and SETKEY modes, and that DROP should have been DELETE as under EDIT.

11. Teachers felt that the procedure of changing the name of a unit was unnecessarily complicated.

12. Most felt that the total number of Mode Two commands should be kept as small as possible. One felt that the WHY command could be obviated by printing the error message whenever a command was "NOT DONE", and that EVALUATE was unnecessary.

Two of the teachers felt that three workspaces were not enough; but the remainder felt that those provided were sufficient (one commented that more would be confusing). Teachers expressed no criticism of the structure of the files, but two additional types of files were suggested. One would be a test bank of multiple choice questions from which teachers or the system could dynamically assemble tests. The other would be a file in which school resources (for example; filmstrips, records, AV Machinery,...) could be accounted for the purpose of providing flexible and dynamic allocation of these to students.

The teachers, having just learned TAIM Mode Two themselves, were asked to suggest a method and the duration of teaching this to others. The majority favored a two-day inservice with roughly half time spent on instruction and discussion and half time spent doing "hands on" work at a terminal with a consultant available. One suggested a

follow-up inservice after two weeks; others felt that one person in the school should take a longer course and later serve as an in-school consultant. Both an instruction manual and a reference manual were advised.

General comments made by the teachers about the Mode Two portion of TAIM included the following.

> It (TAIM) requires that one plan very thoroughly the instructional sequence, and it enables one to construct an optimal instructional sequence... (Pb)
>
> The teacher as programmer must analyze his subject matter closely and have his objectives clearly in mind (Pa).
>
> There is virtually no limit to the extent/variety of 'learning sequences' available through TAIM (Pf).
>
> I am not sure that the system can be made viable if used by only one school. If a number of schools were to use the same materials and if all of the teachers were involved in construction and modifications, then it would probably mean that less time would be required overall (Pb).

CONSTRUCTION OF MATERIALS

The second main task of the participants of the Ed. C.I. 466 class was to prepare eight class days of instruction on logarithms. They were provided with a flowchart of instructional strategy devised by the writer and one of the teachers in the school. (This flowchart is contained in Appendix C.) They were asked about the value of the flowchart as an initial planning device:

> The flowchart technique (or something equivalent) is essential... (Pf)
>
> A flowchart shows the total picture of a unit, the sequence, and the time required... (Pa)

> ... essential in the process of constructing instructional materials, particularly if the content is to be divided among several people (Pd).

> ...an obvious necessity,...(Pb)

> ...provides a quideline for everyone to follow. There are no restrictions on the material that can be included... but (it) does state the minimal to be produced (Pc).

> It may be feasible to have a master flowchart which branches from block to block, and have a sub-flowchart which deals with branching within the blocks (Pe).

The teachers were asked who should have the responsibility for creating and updating the curriculum materials. All felt that the classroom teachers should undertake this job; three suggested that a specialist should be in charge. The problem of classroom teachers not having the time to do this job was recognized; suggestions were: pre-packaged materials available at the outset, preparation done during the summer months, extra teachers to rotate freeing groups of teachers to concentrate on production, and text-entry staff. The participants concurred that regular teachers would be able to handle the updating tasks as part of their regular duties.

In considering the restrictions imposed on Display production by the available computer printer, the teachers made two main points. The first:

> Printer restrictions are very severe. Diagrams and graphs are very difficult to draw; even exponents and subscripts are unnecessarily inconvenient. ...an attempt should be made to introduce more characters into the printer. (Pe)

The teachers' second point was that supporting material was

necessary. Suggestions ranged through teacher-prepared mimeographed materials and overhead transparencies to textbooks designed specifically for use with TAIM, containing mostly diagrams, graphs, tables, and photos.

The teachers had had the experience of "seeing" eight days of material assembled and input during the course of the class. They were asked to outline how they would proceed as an individual teacher with an identical task. All foresaw beginning with an instructional plan for the overall unit (half cited flowcharts). One felt that this was the major task ("Once the master flowchart is drawn, the actual construction of materials is routine (Pe).") Two others gave little emphasis to the construction of Displays and tests ("... make up a lesson plan or outline and then type it onto the terminal (Pc).") Two felt that initial preparation of the content was one of the major tasks. The average amount of time the teachers felt necessary to design and construct the eight days of curriculum materials was forty hours. They estimated a further twenty hours would be needed to enter this material at a terminal.

Three main points were raised by the teachers in the final General Comments sections of their critiques. Some felt that use of TAIM might magnify the dependence of some students on personal teacher help; and that it might also magnify student independence. In either case, this might or might not be a pedagogically "good" thing to have happen.

Some of the teachers maintained that because of the

specification of lessons for students in advance, TAIM can not offer truly individualized instruction.

> (We cannot) determine exactly how to construct each display for each student. ...we are restricted almost to grouping the students into two or three groups based on past performance. Whether past performance is always truly indicative of student needs is debatable (Pb).

The third main point concerned the possibility of incompatability between TAIM and the present lock-step structure of our school system.

2. <u>IMPLEMENTATION RESULTS</u>

The Implementation procedure consisted of using the materials prepared by the Ed. C.I. 466 class with three high school classes. Data was collected from both the teachers and students during this procedure. Achievement pre- and post-tests were given to the students, and they were asked to fill out a questionnaire on the TAIM system. The teachers were given the same Background and Attitude questionnaire as the Ed. C.I. 466 class in both the pre and post situations; at the end of the procedure, a taped interview was conducted by the writer with the teachers.

A) <u>STUDENT ACHIEVEMENT</u>

As mentioned in Chapter Four, it was not the intent of this study to evaluate student achievement under TAIM. However, it did seem desirable to attempt to provide assurances that some learning was taking place. The standard method for doing this is via achievement pre- and post-tests.

For the Math 20 and 33 classes, there was reason to believe that the majority of the students had not studied logarithms before. It was felt that giving these students a long test on material they had not previously encountered would be detrimental to the remainder of the experiment. A short pre-achievement test was considered sufficient, therefore one of only five questions was used.

On the last day of TAIM instruction for each of the three classes, a thirty question final exam was administered. The results of this test were compared to the pre-test scores in two ways.

Considering each item on the pre-test to be worth six marks (5 items, possible score = 30) and each item on the post-test to be worth one mark (30 items, possible score = 30), the results of Table 7 were obtained.

TABLE 7

STUDENT ACHIEVEMENT MEANS: PRE VS TOTAL POST

CLASS	N	Pre-test MEAN	Post-test MEAN
Math 20	31	9.7	18.3
Math 33a	19	12.9	17.0
Math 33b	26	15.5	16.9
All Classes	76	12.5	17.5

The second comparison was made by selecting five questions from the thirty on the post-test as being parallel to the five questions on the pre-test. Considering the two tests to have a possible score of five, the results of Table 8 were were obtained. The achievement pre-test and the final exam, with the selected questions identified, are contained in Appendix C.

TABLE 8

STUDENT ACHIEVEMENT MEANS: PRE VS SELECTED POST

CLASS	N	Pre-test MEAN	Post-test MEAN
Math 20	31	1.6	4.2
Math 33a	19	2.2	3.9
Math 33b	26	2.6	4.1
All Classes	76	2.1	4.1

The above results establish that the students did learn something about logarithms. Although not as a part of the experiment, another Math 20 teacher administered the final logarithms exam to her class of thirty-seven students; she reported a mean of 21.9. This would indicate that her class did learn the subject better, but provision of better instructional materials would very likely improve the learning of the students who used TAIM. This matter is discussed further in Chapter Six.

B) STUDENT COMMENTS

On the last day of instruction for each of the three classes, a student questionnaire was administered. The questionnaire consisted of fifteen questions, the first twelve of which were multiple choice; the last three were open ended. Following are the multiple choice questions. To the left of each possible response is the percentage of students choosing that response (N=69).

1. What name would you suggest for the printouts?
   - 29% a) lessons
   - 12% b) instructions
   - 25% c) guides
   - 28% d) printouts
   - 6% e) other (Please specify) __________
     Responses of e) were: Progress Guides, TAIM, CPOM33, and Compta-Math.

The remaining eleven multiple choice questions had no stem. Students were asked to select the alternative that best represented their feelings or perception.

2. 16% a) I think TAIM should be used for every math class.
   - 64% b) I think TAIM should be used, but combined with regular classes.
   - 16% c) I think TAIM should be used about half of the time.
   - 3% d) I don't think TAIM should be used at all.
   - 1% e) No response.

3. 26% a) I received my own personal printout each day.
   - 25% b) It just seemed that my printout was meant for me personally.
   - 36% c) Everybody received about the same printout.
   - 10% d) I didn't pay much attention to the printouts.
   - 3% e) No response.

4. 6% a) By reading the printouts and text carefully, you could learn the whole chapter.
   - 78% b) With the printouts and text, you only need teacher help once in a while.
   - 13% c) I used the printouts and text, but needed quite a bit of help from the teacher.
   - 2% d) If the teacher hadn't helped me, I wouldn't have learned anything at all.
   - 1% e) No response.

5. 45% a) Having tests almost every day makes me learn better.
   42% b) I didn't mind taking tests almost every day.
   10% c) I don't think we should have had so many tests.
   0% d) Too many tests make it hard to learn.
   3% e) No response.

6. 93% a) I liked getting my test marks back promptly the next day.
   4% b) I wouldn't have minded waiting a few days for my test marks.
   2% c) I didn't really care if I got my test marks back at all.
   0% d) I would rather that only the computer knew my test marks.
   1% e) No response.

7. 61% a) I like multiple choice questions the best.
   24% b) I didn't mind having only multiple choice questions on the test.
   6% c) I would have liked some other types of questions on the test.
   9% d) Multiple choice questions can't really measure what you know.

8. 35% a) All classes should have students' possible sequences of instruction and activities planned in detail in advance.
   19% b) Only math classes should attempt to have sequences planned in advance.
   42% c) Only parts of certain courses should be planned in detail in advance.
   4% d) No courses should have the details of instruction planned in advance.

9. 32% a) I would have liked to have a chart showing the possible sequences of instruction for everyone in the class.
   46% b) I would have liked to have a chart to see what MY possibilities might be.
   15% c) I didn't really care about sequences of instruction.
   6% d) I don't think students should know about the possible sequences of instruction in advance.
   1% e) No response.

10. 22% a) Using TAIM was the best way to learn about logarithms.
    56% b) Using TAIM was as good a way as any to learn about logarithms.
    16% c) Regular classes would have been better for learning about logarithms.
    6% d) There is no good way to learn about logarithms.

11. 36% a) Being a good reader isn't important to using TAIM.
    20% b) If you can't read well, you often have to ask friends or the teacher to explain.
    38% c) Being a good reader would help a lot when using TAIM.
    5% d) If you can't read well, it is very difficult to learn under TAIM.
    1% e) No response.

12. 59% a) With all the tests we took, TAIM could tell just what kind of instruction to give us.
    13% b) The information collected by TAIM wasn't really used very much.
    17% c) The tests didn't really provide TAIM with much information about me.
    2% d) I don't think TAIM should have all that information about me.
    9% e) No response.

Question 13 of the student comments questionnaire asked students to write the two things they most liked about TAIM. To this question, a large number of students gave responses indicating they appreciated the daily feedback provided by the system. Some examples of this type of comment are:

> Getting test results back fast so you can see where you went wrong.
>
> I liked to know what I had learnt each day.

A large number of responses were given which indicated students enjoyed working on their own. These appeared to have resulted for several different reasons as the following examples show.

> Made you figure out more things by yourself.
>
> A person could see if he can learn without anyone pushing him.
>
> Gave the experience to the individual about self learning maybe for the future.

Another large class of responses indicated that students liked the informal classroom atmosphere and the availability

of the teacher for personal assistance.

> The easy atmosphere of going ahead and doing your own work.
>
> The teacher and the computer had more time to deal with your problems.
>
> There is a relaxed feeling. That is you're in no hurry or being pushed into anything.
>
> You don't feel so stupid asking a teacher a question when you've spent time trying to hash it out with someone else first.
>
> If you had any problems, you could get individual attention.
>
> The teacher being able to help when we needed it.

Two other frequent types of response indicated that students felt they were working at their own rate and that the system made the subject easy to learn.

> I guess the main thing I liked about TAIM was that you could progress at your own rate.
>
> It was quite easy to follow, it added a change to Math. It almost made it fun.
>
> It was a much easier form of learning that daily teacher work.

Some interesting further responses to question 13 were as follows.

> The way we were programmed into different day programs according to our previous days marks
>
> We were allowed to take time over what you were reading and think over it.
>
> Learned faster, you know just how smart you are.
>
> A logical progression that I could follow.
>
> The computer had your lesson set out well in advance.
>
> Working within my own capabilities.

> It gave a variety of things to do each day and gave better understanding as how to do them.
>
> Being on a set course to arrive at a certain goal.
>
> By working sort of on your own you get a better idea of what you do know and what you don't.
>
> It stops you, makes you review, and then proceeds when you are ready.
>
> I liked getting right into the lesson, and not having to listen to the teacher explain to others.
>
> Only the necessary was there. Everything you need was there without having too much.
>
> When you came to class, you knew what to do and did it.
>
> The tests on each section told me if I was learning enough. (Its better to have tests like that every day than to have only a chapter test at the end & not review what you missed.)

Question 14 of the student comments questionnaire asked students to write the two things they most disliked about TAIM. About one quarter of the students did not respond at all to this question. Of the responses received, the largest category expressed that students felt "rushed" at some time.

> Some of the lessons were too long and I didn't have time to finish.
>
> I felt a bit rushed on 1 or 2 days when I wanted to review or take my time but couldn't because I had to finish.

This latter comment also fits in a second category of responses. Students objected because they felt that the system required full time attendance and required them to master each day's lesson.

> If you missed some classes on TAIM, you would have to do a lot of catching up.

I missed 3 days. There didn't seem to be any way for me to catch up on work except for reading the book.

If you miss a point or can't follow, TAIM has no way of knowing.

It required, on the better part, full attendance.

Further comments made by individual students were:

I couldn't tell the difference between division and subtraction from the printouts.

Typing i.e. 132/141

The huge sheets of paper get in the way.

Wasted too much paper.

Learning that someday we'll be run by Computers.

It made me feel as if I were just an I.D. card, not a person.

Although logical, it is useless when applied in a teaching situation, as infinite personal responses require an infinite number of computerized responses.

In class we worked things out by approximating and in TAIM you worked things out exactly.

It covered information we should do in class - this makes the teacher look lazy.

The way this project was presented there was no need for any class period. Personally, I was annoyed at the fact that we were still held in a systematic atmosphere. I had no teacher instruction for this thing and would have rather just had the copies, done them on my own time and when I felt like it. The project was good in itself, but not presented in a good way by authority.

<u>Question 15</u> was as follows:

Suppose that you had a friend who knew nothing about TAIM. Write a description of the system you might give to him. Include all of the things you think are important. You may use point form.

The purpose of this question was to attempt to obtain a picture of the TAIM system from the students' point of view. From the numerous points listed by the students, examples were selected and categorized; these are reported below.

1. <u>System Structure</u>

   With this method, you use a computer printout, the text, and the teacher. Giving a complete instruction method.

   You get large sheets of paper which include pages of reference showing what to read; shows you pages where good examples are illustrated; gives you exercises in the textbook for practice.

   It is a good system for learning on your own. Each day there is a quiz and the computer makes out your program for the next day based on your quiz.

   You teach yourself using the text and the guides only asking your instructor for help if all else has failed.

2. <u>General Procedures</u>

   Each day you get a different lesson.

   Printouts would be distributed to everybody and then everyone was left to progress on their own.

   There are lots of examples and everything is explained fully.

   Requires the use of a textbook and a little oral instruction.

   An exam was included at the end of every lesson.

   You keep all printouts for your own use.

   You have little or no assigned homework.

3. <u>Quiz Procedures</u>

   You would write a little quiz almost every day and you get your results back the next day.

   You put all answers in one form on an answer card.

   You get results next day with right answers

available.

You know your own result without everyone else knowing.

4. <u>System Procedures</u>

Write tests atout every day sc computer can decide to advance you new material or to give reviews.

If you get a good mark on the first test and somebody else got a rotten one, you and the other person would get different tests.

Judging by your marks the computer decided your personal lesson for the next day. A certain amount of review if necessary or a brand new lesson.

5. <u>Classroom Procedures</u>

We read the lessons over and go on and do the work suggested.

It tells you, in your own program, what page to turn to, and what it is all about.

The tests (short) at the end of each printout was just a short way of summarizing what you had just taken.

If a student had any trouble he was able to ask a friend and if he was really stuck he could consult with the teacher.

Learn through reading and comprehending what you have in front of you rather than being told this & this and having to accept it.

C) <u>TEACHER QUESTIONNAIRE</u>

The Background and `Attitude questionnaire was administered to the two teachers involved in the Implementation procedure. As with the Ed. C.I. class, the Attitude portion was administered in both the pre and post situations. The results of the Background portion indicated that both teachers had taken one course involving computers

and that both had learned one interpretive computing language. (From discussions it was found that this was the BASIC language learned while a PDP-8E computer had been on loan to their school.) Both had four years of University education; one had taught for six years, the other for four. Their scores for the Attitude portion of the questionnaire are reported by category (as described in part A of the previous section - see also Table 1) in Table 9.

TABLE 9

IMPLEMENTATION ATTITUDE QUESTIONNAIRE SCORES

		CATEGORY ONE SCORES				CATEGORY TWO SCORES				CATEGORY THREE SCORES			
TEACHER		1	2	3	4	1	2	3	4	1	2	3	4
Ta	pre	1	1	4	2	1	1	4	2	0	5	3	1
	post	0	1	5	2	3	4	1	0	2	2	4	1
Tb	pre	2	0	0	6	1	3	3	1	2	2	4	1
	post	0	2	2	4	2	1	3	2	1	1	5	2
TOTALS													
	pre	3	1	4	8	2	4	7	3	2	7	7	2
	post	0	3	7	6	5	5	4	2	3	3	9	3

D) <u>TEACHER INTERVIEW</u>

On the day after completion of using TAIM with the Math 33 classes an interview was conducted with the two teachers and Dr. Kieren. The purpose of the interview was to compile the teachers' reactions to the TAIM system. The session was recorded on tape. The teachers felt that the main advantage

of using TAIM over regular instructions was that they were able to spend most of their time working with individual students. They were concerned that a low reading ability might hamper some students' progress: but noted that the major part of the material presented was simple enough for everyone to read; that those who were having difficulty had ample opportunity to seek clarification from a fellow student or the teacher; and that if TAIM were used for an extended period of time it would likely motivate and help develop better reading skills. (A study by Sunde, 1970, showed a non-significant correlation between reading ability and achievement in an environment similar to that created in this experiment.)

The teachers further felt that TAIM provided good preparation for University; giving the students an opportunity to learn on their own and to do independent study not encouraged by regular instruction. Even over the short interval of the implementation, the teachers felt that students had aquired a better ability to study and review material; and that they developed skills of learning from each other.

One of the teachers taught a section on logarithms to a class not using TAIM after the first eight day session of the experiment. He commented that he was teaching better now because of the experience of using TAIM; but felt that his students weren't learning the concepts as well. He conceded that this feeling might be due to him simply paying more

attention to individual difficulties. Both teachers felt that using TAIM had made them more conscious of the detail of the curriculum they were teaching and the various little difficulties that students might encounter.

> TAIM did not help all students. Two or three students taking the program did change their marks, making them higher or lower. Students who were doing 65 to 70 suddenly became 45s; 55 turned out to be 75 or 80. In other words, for some kids it was very advantageous to take it - for some not (Ta).

The only disadvantage of the system compared to regular instruction that the teachers were able to identify centered on the teachers' ability to extemporaneously revise the instruction. (This point is discussed in detail in part A, section 2, Chapter Two.) However, they were aware of their power to revise the curriculum kept by TAIM. Discussion on disadvantages centered on the teachers regretting that they did not have an opportunity to go through the loop of revising and trying out their new materials several times. They felt they could have produced a much better set of materials, and indicated that they would have included many more alternatives.

> ... if we'd had a chance to go through it completely and watch it run; then go back and correct those things that we wanted to change. We could've added more parts, thrown out more parts. As a management system, I thought it was great. I really enjoyed working with it (Tb).

The teachers learned and used the Mode One command set. They either identified or agreed with the following proposed alternatives:

1. A facility for obtaining class statistics (average,

standard deviation, etc.) on a particular test. (This was planned for the MARK command but not implemented.)

2. A facility for quick identification of students for whom a card had not been read. In the implemented version of TAIM, this was possible only by obtaining a TRACE or MARK across the entire class.

3. Enhancement of the READ command so that (perhaps optionally) a student's test score and next-lesson pointer would be printed as soon as his marks were read. (This too, was planned but not implemented.)

The main further addition that the teachers identified was that it should have been possible to provide little Displays for each test question a student got wrong to attempt to correct his difficulty. This is not easy to accomplish with the current system design.

A further point made by the teachers is illustrated below with the actual dialogue. Unattributed statements are those of the writer.

> The card reader... it does take a lot of time (to type in the responses) (Ta).
>
> Even a card reader and line printer wouldn't hurt (Tb).
>
> A card reader would cost about ten dollars a day.
>
> But we were running in a neighborhood of five to six dollars every time (Tb).
>
> I was thinking of the cost of the teachers... a terminal which will support the card reader and print 250 lines a minute costs around $10,000. One like that would handle quite a number of teachers...
>
> Well it wouldn't be that expensive if you were to

> run TAIM for the entire year. The extra time alone would almost pay for it in a year (Tb).
>
> How many teachers would want to learn how to run a terminal like that and learn TAIM?
>
> I think most would; at least those in Math and Science (Tb).
>
> I think those would be the two disciplines involved ... at least immediately (Ta).

Some time was spent in discussion of the use of TAIM in other subjects - mainly Social Studies, but no firm conclusions were reached except that the teachers could envision several alternatives. If the system came to be used widely, it was felt that the Department of Education or some other provincial body might take the leadership in providing a public library of Displays and Tests - and a catalog describing these. The teachers felt that a single teacher with the assistance of an aide would be able to provide good individualized instruction to a class of up to one hundred students or more using TAIM, and that rearrangements of teacher duties under this structure would enable teachers to devote the time needed to constructing and revising curriculum. The teachers' view of the present structure after using TAIM is illustrated by the following dialogue:

> One of the students commented that they didn't see why they had to stay in the classroom (while using TAIM).
>
> I agree; she's got a point too. But its just the way our system is structured (Tb).
>
> Its her wish to attend classes when she wants to (Ta).
>
> I think she's in the wrong type of system (Tb).

3. COST ACCOUNTING

The costs of operating the TAIM system arise in a number of areas; some are directly measurable and others are not. Part of the cost of the experiment has already been indicated by the time teachers spent in learning TAIM and in preparing curriculum materials. The purpose of this section is to present the remainder of the cost data collected. This includes the statistical records generated by the MONITOR and the DRIVER, the actual costs of running the DRIVER, and some Hardware cost estimates. The reader is cautioned that the data presented in the first two parts of this section is completely dependent upon the experimental implementation and is therefore nct useful for the purpose of direct comparisons to other types of instruction. The data and the restrictions that apply are discussed in section three of Chapter Six.

A) SYSTEM STATISTICS

Both the MONITOR and DRIVER programs produced statistical records for each command or statement interpreted. The details of the structure of these records might more properly have been included in Chapter Three. They are included here to indicate exactly what type of data was collected. Analysis of these records resulted in Tables 11 through 14.

TABLE 10.

FORM OF STATISTICS RECORDS

ABBRE-VIATION	FIRST BYTE	NUMBER OF BYTES	FORM	NAME
ID	1	6	characters	IDENTIFICATION
DATE	7	8	YY-MM-DD	DATE
TIME	15	8	HH-MM-SS	TIME
T/B	23	1	0 / 1	TERMINAL/BATCH
CPU	24	5	1/1000 sec	CPU TIME
EL/L	29	5	secs/number	ELAPSED/LINES
VMI	34	5	core pages	VIRTUAL MEMORY
COST	39	5	cents	COST
MODE	44	1	0,1,2,3	MODE
CMD	45	2	2 thru 39	COMMAND NUMBER
ERR	47	3	0 thru 415	ERROR CODE
TEXT	50	variable	text	COMMAND TEXT

ID: For MONITOR statistics, this field contained the user's six character TAIM identification code. For DRIVER records, this contained the student identification code.

DATE: The date of issuance of the command, where YY is the year, MM is the month and DD is the day.

TIME: The time of issuance of the command, where HH is the hour, MM is the minute and SS is the second.

CPU: The total cpu time in thousandths of a second required to execute the command.

EL/L: For MONITOR records, the elapsed time in seconds between request for and completion of the current command. For DRIVER records, the number of lines printed.

VMI: The size of the core space in use at the completion of the command in pages (one page = 4096 bytes).

COST: For MONITOR commands, the dollar cost was calculated as 300c + 3cv + 1.5e where "c" is the CPU time in hours, "v" is the virtual memory integral, and "e" is the elapsed time in hours. This was all multiplied by 1.3 (rate factor) if processing was online. For DRIVER commands, costs of generating student printouts were

retrieved from a system subroutine and an amount added to cover the cost of printing.

MODE: For MONITOR records 1, 2, or 3 represented the mode from which the command was issued. For DRIVER records, 0 was used.

CMD: Each MONITOR command has a command number ranging from 1 through 39. Each logic statement was numbered from 1 through 6 and as well, a HOLD produced a statistic with CMD of 7, a NOTE with a CMD of 8; and a TOTAL record with CMD of 9 was produced for each student processed by the DRIVER. (This record contained the total CPU, PAGES, and COST for producing the students' printout.)

ERR: Each MONITOR command could be not executed because of an error condition. These conditions were numbered and were reported in this field. If the command was successfully executed the ERR was 0; if suspended by an ATTENTION interrupt the code was 415. The remaining codes depended upon the error condition raised.

TEXT: For the MONITOR, the text of the entered commands; for the DRIVER, the text of the LOGIC statement or the HOLD or NOTE record or '**TOTAL**'.

TABLES 11, 12, 13, and 14 provide breakdowns of the costs of processing the logic statements, and the three modes of MONITOR commands. In these tables, ONL/BTCH indicates whether the command was processed on-line from a terminal (ONL) or off-line in a batch environment (BTCH). ISSUES indicates number of times the command or statement was encountered; OK the number of successful attempts and ERR the number resulting in errors (the results from commands raising errors were not included in the calculations of the remainder of the values). MAX, MIN, AVG, and STD represent maximum, minimum, average, and standard deviations of the values.

TABLE 11

LOGIC FILE STATEMENT STATISTICS

COMMAND	ONL BTCH	ISSUES OK	ISSUES ERR	CPU TIME (SECONDS) MAX	CPU TIME (SECONDS) MIN	CPU TIME (SECONDS) AVG	CPU TIME (SECONDS) STD	NUMBER OF LINES MAX	NUMBER OF LINES MIN	NUMBER OF LINES AVG	NUMBER OF LINES STD	AVG VMI	COST (DOLLARS) MAX	COST (DOLLARS) MIN	COST (DOLLARS) AVG	COST (DOLLARS) STD
IF	BTCH	340	0	0.14	0.01	0.05	0.03	0	0	0	0	80	0.02	0.01	0.01	0.00
MESG	BTCH	53	0	0.06	0.04	0.05	0.01	0	0	0	0	81	0.01	0.01	0.01	0.00
SHOW	BTCH	1153	0	1.27	0.12	0.60	0.30	163	14	83	39	79	0.26	0.02	0.12	0.06
TEST	BTCH	591	0	1.34	0.18	0.47	0.32	114	29	56	28	79	0.22	0.04	0.09	0.05
HELD	BTCH	30	0	0.20	0.15	0.17	0.01	9	9	9	0	80	0.04	0.03	0.03	0.00
NOTED	BTCH	330	0	1.17	0.05	0.36	0.29	132	2	29	38	80	0.19	0.01	0.06	0.05
TOTAL	BTCH	863	0	12.40	0.11	1.38	0.63	1786	7	164	92	79	2.22	0.02	0.26	0.12

TABLE 12: MODE ONE COMMAND STATISTICS

	ONL	ISSUES		CPU TIME (SECONDS)				ELAPSED TIME				AVG	COST (DOLLARS)			
COMMAND	BTCH	OK	ERR	MAX	MIN	AVG	STD	MAX	MIN	AVG	STD	VMI	MAX	MIN	AVG	STD
NOTE	ONL	38	3	0.34	0.03	0.09	0.05	413	8	75	101	89	0.26	0.01	0.06	0.06
READ	ONL	45	0	7.02	0.07	1.99	2.13	1341	23	311	332	88	1.91	0.04	0.57	0.60
HOLD	ONL	1	0	0.03	0.03	0.03	0.0	7	7	7	0	101	0.01	0.01	0.01	0.00
MARK	ONL	57	40	5.47	0.04	1.22	1.53	462	8	100	108	91	1.24	0.01	0.30	0.36
	BTCH	14	0	7.55	1.86	3.77	1.49	39	8	19	9	97	1.24	0.31	0.63	0.24
MODE	ONL	17	2	0.08	0.02	0.03	0.01	218	2	24	51	88	0.14	0.01	0.02	0.03
	BTCH	25	0	0.01	0.00	0.00	0.0	1	0	0	0	99	0.00	0.00	0.00	0.00
POINT	ONL	42	11	2.32	0.03	0.45	0.64	277	6	42	55	90	0.50	0.01	0.12	0.14
DISPLAY	ONL	61	13	1.59	0.02	0.21	0.25	108	4	27	34	92	0.32	0.01	0.06	0.06
	BTCH	3	0	0.61	0.36	0.44	0.12	2	1	1	1	101	0.11	0.06	0.08	0.02
REGISTER	ONL	143	24	0.56	0.14	0.23	0.06	173	19	40	21	92	0.15	0.04	0.07	0.02
UNREGSTR	ONL	6	1	0.20	0.08	0.13	0.05	42	13	25	13	95	0.06	0.03	0.04	0.01
TRACE	ONL	49	35	2.13	0.05	0.52	0.66	308	9	84	101	94	0.59	0.02	0.15	0.19
	BTCH	12	0	9.54	0.06	1.71	2.47	38	0	7	9	104	1.57	0.01	0.29	0.41
WHO	ONL	13	6	1.68	0.05	0.77	0.64	110	6	29	30	91	0.37	0.02	0.17	0.14
WHY	ONL	45	0	0.06	0.02	0.05	0.01	40	1	6	7	89	0.03	0.01	0.01	0.01
UNNOTE	ONL	12	1	0.05	0.03	0.04	0.01	30	5	11	7	85	0.02	0.01	0.01	0.00
AVERAGE	ONL	2	14	9.27	0.23	4.75	4.52	1073	294	683	390	91	2.41	0.21	1.31	1.10
	BTCH	7	0	7.27	3.32	4.87	1.33	62	16	29	16	94	1.20	0.55	0.80	0.22

TABLE 13

MODE TWO COMMAND STATISTICS

	ONL	ISSUES		CPU TIME (SECONDS)				ELAPSED TIME				AVG	COST (DOLLARS)			
COMMAND	BTCH	OK	ERR	MAX	MIN	AVG	STD	MAX	MIN	AVG	STD	VMI	MAX	MIN	AVG	STD
DROP	ONL	23	10	0.69	0.22	0.50	0.11	73	7	18	18	92	0.16	0.06	0.12	0.03
EDIT	ONL	168	11	5.86	0.22	0.49	0.48	2585	11	112	218	91	2.60	0.06	0.17	0.21
EVALUATE	ONL	20	4	0.66	0.02	0.28	0.24	36	4	10	7	90	0.15	0.01	0.07	0.05
FIND	ONL	13	1	0.47	0.07	0.27	0.17	40	7	11	9	96	0.12	0.02	0.07	0.04
LABEL	ONL	136	24	0.23	0.02	0.06	0.04	97	4	18	16	90	0.07	0.01	0.02	0.01
MODE	ONL	46	5	0.03	0.02	0.02	0.00	36	2	7	6	88	0.03	0.01	0.01	0.00
	BTCH	4	0	0.01	0.01	0.01	0.0	0	0	0	0	99	0.00	0.00	0.00	0.00
PLACE	ONL	130	20	1.46	0.05	0.55	0.32	190	5	21	23	93	0.33	0.02	0.13	0.07
WHY	ONL	76	0	0.07	0.04	0.05	0.01	77	2	6	12	82	0.05	0.01	0.01	0.01
SAVE	ONL	145	18	10.12	0.32	1.85	1.59	99	5	20	15	91	2.27	0.07	0.40	0.35
SETKEY	ONL	15	5	1.68	0.51	1.02	0.35	515	79	240	139	88	0.62	0.16	0.34	0.13
MTS	ONL	27	0	2.09	0.06	0.63	0.61	121	7	27	23	92	0.53	0.02	0.15	0.15
CATALOG	ONL	92	28	1.61	0.03	0.30	0.37	324	6	43	48	82	0.42	0.01	0.09	0.10

TABLE 14

MODE THREE COMMAND STATISTICS

COMMAND	ONL BTCH	ISSUES OK	ISSUES ERR	CPU TIME (SECONDS) MAX	CPU TIME (SECONDS) MIN	CPU TIME (SECONDS) AVG	CPU TIME (SECONDS) STD	ELAPSED TIME MAX	ELAPSED TIME MIN	ELAPSED TIME AVG	ELAPSED TIME STD	AVG VMI	COST (DOLLARS) MAX	COST (DOLLARS) MIN	COST (DOLLARS) AVG	COST (DOLLARS) STD
SET	ONL	1	0	0.04	0.04	0.04	0.0	9	9	9	0	70	0.01	0.01	0.01	0.00
WHY	ONL	5	0	0.05	0.04	0.05	0.00	3	2	2	2	84	0.01	0.01	0.01	0.00
REGISTER	ONL	19	4	0.16	0.10	0.12	0.02	47	12	26	10	75	0.05	0.03	0.04	0.01
UNREGSTR	ONL	1	0	0.08	0.08	0.08	0.0	16	16	16	0	70	0.02	0.02	0.02	0.00
MODE	ONL	55	C	0.05	0.00	0.02	0.00	62	0	5	9	88	0.04	0.00	0.01	0.01
	BTCH	13	0	0.01	0.00	0.01	0.0	1	0	0	0	102	0.00	0.00	0.00	0.00
DISPLAY	ONL	24	10	0.21	0.13	0.16	0.02	74	39	48	9	72	0.08	0.05	0.06	0.01
	BTCH	4	0	0.18	0.07	0.12	0.06	1	0	0	0	104	0.03	0.01	0.02	0.01
INITIAL	ONL	6	2	0.40	0.06	0.20	0.15	33	5	11	10	83	0.09	0.02	0.05	0.03
OFFLINE	ONL	1	0	0.20	0.20	0.20	0.0	55	55	55	0	67	0.07	0.07	0.07	0.00
MTS	ONL	3	0	0.88	0.22	0.45	0.31	386	81	198	134	67	0.37	0.08	0.19	0.13

[illegible]

In Table 11, the values cited for the _IF_ statement execution comprise the CPU time and cost of evaluating the condition. If the Condition was true and a subsequent SHOW or MESG was required, these resulted in separate statistics records. If a student was _HELD_, the statistics record produced accounted for the entire cost of processing the student.

Across from _TOTAL_ is a summarization of records that were produced for each student in each run indicating the total cost of processing that student. The values of MAX under CPU time, LINES, and COST are a little misleading. These values resulted from a single occurrence where a "dummy" student was branched to a "dummy" logic unit which requested a printout of all of the available Tests and Displays. The next largest maximum for these three are 2.76, 359, and $0.47 (respectively).

DRIVER COSTS

When running any program in batch mode at the University of Alberta, a "Tail-Sheet" is produced by the system detailing the costs of the run. These tail-sheets of the DRIVER runs were kept and are the main source of data for this section. For each run of the DRIVER, the total cost of the run was itemized on the tail sheet in five categories: CPU time, virtual memory, lines printed, pages printed, and tape drive use. Tape drive use amounted to only one percent of the cost and so was not analyzed. For all of the DRIVER runs, use of the computer (CPU time plus virtual

memory) amounted to roughly four fifths of the cost; use of the printer (lines and pages) amounted to one-fifth.

The charge for Virtual Memory made at the University of Alberta is calculated by taking an Integral of the function of amount of computer core storage being used over the amount of CPU time used. While the CPU time used for the various parts of the DRIVER run was kept, the value of the Virtual Memory Integral was not available. However, since the amount of virtual memory in use at any instant in the processing by the DRIVER was assumed to be constant from run to run, the value of the Integral was assumed to be directly proportional to the amount of CPU time used. This assumption is the the basis for the construction of Table 16. Because of the overriding effect of CPU use in the determination of the costs of using TAIM, a presentation of its allocation by category is presented in Table 15.

A total of eighteen <u>RUNS</u> producing printouts for the students were executed. Tables 15 and 16 show only sixteen of these. The first run is not included because the procedures for maintaining backup were not implemented at that time (in fact it was due to a failure in attempting to process the first run that the nature and extent of the backup was determined). Run two is omitted because during that run a malfunction of the MONITOR caused the logging of approximately fifty seconds of non-productive CPU time (it got caught in a loop). Run ten was done solely to obtain the results of some off-line MONITOR commands. One page was

printed for each student with their name and little else. This run has been kept because it provides a "base-line" instance of DRIVER use.

TABLE 15.

CPU TIME BY CATAGCRY OF DRIVER OPERATION

RUN	BACKUP	DRIVER PROCESS	DRIVER SORT	MONITOR	OVERHEAD	TOTAL
MATH 20 (n=37)						
3	15.4	61.5	26.8	8.2	25.9	137.8
4	18.9	56.3	36.3	6.6	14.5	132.6
5	29.1	80.6	53.1	8.1	18.6	189.5
6	25.1	55.1	32.1	15.1	18.5	145.9
7	24.6	47.9	22.5	5.8	18.2	119.0
8	41.6	73.5	35.6	0.5	20.2	171.4
9	29.6	50.3	28.5	0.4	17.1	125.9
10	25.1	21.6	8.0	29.3	17.8	101.8
AVG	26.2	55.9	30.4	9.3	18.9	140.5
MATH 33s (n=53)						
11	17.5	51.9	32.9	2.0	19.1	123.4
12	19.9	56.6	31.8	0.4	18.4	127.1
13	24.0	70.1	37.5	5.4	17.9	154.9
14	31.1	98.0	53.7	2.8	18.4	204.0
15	32.5	125.9	67.2	8.6	18.6	252.8
16	26.8	74.2	40.8	0.5	18.4	160.7
17	25.2	74.0	28.2	0.4	16.6	144.4
18	34.2	93.0	57.0	30.0	19.4	233.6
AVG	26.4	80.5	43.6	6.3	18.4	175.1

All entries are in seconds.

The Math 20 and 33 classes are listed separately. This was done mainly because of the different number of students involved in the two cases but also because the curriculum was changed. In terms of quantity of output per student, it is meaningful to compare runs 4 with 14, 5 with 15, and 6 with 16 for the different groups.

The <u>BACKUP</u> category in these tables includes: the CPU

time or COST attributable to maintaining necessary tape backup in case something went wrong and some miscellaneous routines that provided statistics about how other routines were proceeding. These statistics-generation routines might be more properly categorized as preventative maintenance procedures.

TABLE 16.

COST BY CATEGORY OF DRIVER OPERATION

		DRIVER				
RUN	BACKUP	PROCESS	PRINT	MONITOR	OVERHEAD	TOTAL
MATH 20 (n=37)						
3	$2.16	$12.37	$4.22	$1.15	$3.63	$23.53
4	2.45	11.99	5.55	0.85	1.88	22.71
5	3.88	17.82	7.37	1.08	2.48	32.63
6	3.40	11.81	4.51	2.05	2.51	24.27
7	3.28	9.38	3.55	0.77	2.42	19.40
8	5.49	14.39	4.79	0.07	2.66	27.40
9	3.87	10.31	3.51	0.05	2.24	19.99
10	3.95	4.66	1.01	4.62	2.80	17.04
AVG	$3.56	$11.59	$4.31	$1.33	$2.59	$23.37
MATH 33s (n=53)						
11	$2.35	$11.37	$5.10	$0.27	$2.56	$21.64
12	2.65	11.78	4.81	0.05	2.45	21.75
13	3.25	14.55	5.99	0.73	2.42	26.93
14	4.21	20.51	8.05	0.38	2.49	35.64
15	4.49	26.70	9.49	1.19	2.57	44.44
16	3.57	15.31	6.18	0.07	2.45	27.57
17	3.37	13.68	4.23	0.05	2.22	23.56
18	4.79	21.03	6.87	4.21	2.72	39.61
AVG	$3.59	$16.87	$6.34	$0.87	$2.48	$30.14

The production of the students lessons by the DRIVER is done in two stages. First the records are generated, and accounting for this is shown under the heading PROCESS. Before printing however, these records must be sorted by

class and student. In table 15, the CPU time attributable to the sort step is reported separately under the heading _SORT_. In Table 16, both the generation and sorting costs are included under _PROCESS_, but the cost of printing out the lessons (i.e., the charges for pages and the actually printing of lines) is reported under _PRINT_. The CPU time or charges attributed to the running of the MONITOR are listed under _MONITOR_, and the total CPU time and the cost of the run as reported on the tail-sheets is listed under _TOTAL_. Almost all runs of the DRIVER were done during the late evening and were therefore able to utilize a RATE FACTOR (see Appendix E) multiplier of 0.7 on the cost of the run. However the effect of the rate factor has been removed from Table 16 to standardize the values and to give a more accurate picture of the actual costs.

In both tables, the column labelled _OVERHEAD_ was calculated as the difference between the TOTAL and the sum of the other four values for each run. This item consists mainly of the CPU time (and associated cost) necessary to load the routines to be run from disc into core. It also includes costs of some processes more properly categorized under BACKUP but for which CPU times were not available. For one run (number 16) special measures were taken to obtain a breakdown of the overhead. Of the total 18.4 seconds, 12.6 were attributed to the loading of routines (4.5 for loading the DRIVER and 4.9 for loading the MONITOR) and 5.8 to other system operations.

In the construction of Table 16, costs were assumed to arise from only two sources, use of the computer and use of the printer. The number of lines produced by the DRIVER and the number of pages printed for the students were known for each run enabling the calculations of PRINT as the printing cost solely attributable to the printing of lessons. Removing this item from the TOTAL cost, the remainder was aportioned to the other categories as a function of the CPU usage.

C) OTHER COSTS

Two further costs incurred as a result of using the TAIM system may be categorized as Disc Storage and Hardware.

Under the MTS system at the University of Alberta, the unit of measure of disc storage is the disc-page. A disc-page will hold 4096 bytes of data or approximately two double-spaced typewritten pages of text. This is charged (see Appendix E) at the rate of $.50 per page-month. Table 17 shows the size and corresponding costs of the major files kept. Note that the STUDENT file grew from a minimum of ten pages at the start of the experiment to thirty-eight pages as student records accumulated.

At the end of the experiment, TAIM was actually using a total of 214 pages. The difference was distributed as: 66 pages for the MONITOR object, 20 pages for the DRIVER, 6 pages for the HOLD and NOTE files, 10 pages for utility programs and 4 pages for workspaces. The MONITOR, DRIVER, and utility routines could have been kept on tape as they

did not require random access. They were kept on disc merely as a convenience.

TABLE 17.

DISC STORAGE CHARGES

FILE NAME	NUMBER OF LINES	DISC PAGES	FILE CONTENTS	COST PER MONTH
DISPLAY	1785	38	20 DISPLAYS	$19.00
LOGIC	138	4	20 LOGIC UNITS	2.00
STUDENT (min)	328	10	STUDENT RECORDS	5.00
(max)	1380	38		19.00
SYSTEM	424	11	FIXED SYSTEM DATA	5.50
TEST	463	12	9 TESTS	6.00
USER	225	5	REGISTRATION & STATS	2.50
TOTALS (min)	3363	80		$40.00
(max)	5415	108		54.00

The hardware used in the school consisted of two pieces of equipment: a Texas Instrument Silent 700 portable terminal and an Omnitec accoustical coupler (or data set). This equipment was purchased in 1972 by the Division of Educational Research Services for approximately $3,300. In the spring of 1973, it sold for $2,865. The company currently distributing this equipment was not prepared to rent it, but rental can be estimated fairly accurately at $100 per month.

The use of the Omnitec coupler was not entirely satisfactory. The connection between the terminal and the computer was frequently difficult to establish, and in several instances was lost during transmission. A good deal

of this difficulty was thought to have been caused by inadequacies of the telephone line. The problem would likely have been resolved by the installation of an Edmonton Telephones 10382 data set (or MODEM) and a higher grade telephone line. The total cost for installation of these would have been $32. The rental would have been $41 per month.

The teachers typically spent up to a half hour each day simply entering the students responses. Since the system is supposed to remove clerical-type duties from the teachers job, this was unacceptable. Minimum equipment to circumvent this task was cited by IBM as consisting of either a 2740 or a 1050 terminal supporting a 2956 optical mark card reader. Such equipment would operate properly over the higher grade telephone line through the data set. The rental quote from IBM for both of these was $249.09 per month (this includes a discount applied to equipment to be used for educational purposes).

Use of the optical mark reader would mean that students could place their responses onto special cards to be read directly by the equipment. After an initial outlay of approximately $80 to prepare a master plate these cards could be produced for less than $1 per thousand.

The costs quoted above were current at this writing, but in general tend to decrease with time. While it is easy to conceive of more elaborate equipment in the school, TAIM was designed to make heavy use of central facilities during

the late evening hours, and to require only a minimum configuration at the school.

The data presented in this chapter was derived from the procedures outlined in Chapter Four. Although it is applicable only to the prototype TAIM System, it does provide a basis from which extrapolation can be made, and grounds for criticizm of the prototype. This is the topic of Chapter Six.

CHAPTER VI

SYSTEM FEASIBILITY AND RECOMMENDATIONS

The purpose of this study was to develop a prototype of the TAIM System and to conduct a feasibility assessment of it. The Preparation procedure, which consisted of training six people to use Mode Two of TAIM and having them prepare curriculum, has provided data on the effectiveness of TAIM as a vehicle for the preparation and storage of instructional sequences and materials, and the training of teachers in the use of the system. The Implementation procedure, the use of the system with the prepared curriculum in a school, has provided indications of the effectiveness of TAIM as a manager and instructional assistant. Cost data collected across both procedures are an integral part of the feasibility assessment.

As an experiment with the process of instruction, this study has raised a number of questions concerning pedagogy and philosophy of education. Further issues such as man-machine interaction and the matching of the TAIM System to the model of instruction presented in Chapter Two remain to be examined. However, the feasibility assessment is seen as a first priority requirement and is the central topic of

this final chapter. The data of Chapter Five are herein examined and further data in the form of observations and conjectures by the writer are introduced. The chapter and the study are concluded with recommendations for the further use of the TAIM System.

1. PREPARATION FEASIBILITY

The Preparation procedure consisted of training six teachers-in-training to use TAIM; and having them design, devise, and input a segment of curriculum to the system. While these teachers were exposed to the entire TAIM system the emphasis was on learning and using only the Mode Two portion. Correspondingly, the assessment presented here focuses on that part of TAIM and that part of the role the teacher concerned with the design of and preparation for instruction.

A) MODE TWO

THE MODE TWO COMMAND SET

The Mode Two command set, as implemented, allowed the teachers to accomplish the jobs that were required. However, as a result of the Preparation procedure, several modifications would seem to be desirable. The following list of suggestions concerning the commands are arranged in decreasing order of importance.

1. A number of factors resulted in making the use of the EDIT facility under TAIM too costly. In an effort to reduce costs, the teachers were encouraged to use the MTS

File Editor outside of TAIM. This resulted in a need to master a number of procedures described as "peripheral" under the presentation of the TAIM Principles Exam discussed in Chapter Five. TAIM was designed to operate without such peripheral requirements. Although the possibility of such external operations should not be removed, elimination of the necessity for such actions is seen as a first priority requirement for improvement of the system. The most direct and effective means of accomplishing this would be to make the MONITOR code "re-entrant;" that is, capable of being executed by more than one user at a time. Having done this, the MONITOR could be loaded once and then used by a number of users simultaneously; resulting in the EDIT mode being no more costly than use of the System Editor.

As a further extension to this suggestion; TAIM was considered in its design to be an operating system; that is, the core resident program in control of the computer. While the implemented version is not capable of controlling a computer at present, a dedicated CPU and peripherals are seen as the most economically feasible method of using TAIM with a large number of teachers and students.

2. TAIM imposes a number of restrictions on when a user may or may not alter the disposition of units. These restrictions ensure that the integrity of the permanent files is constantly maintained, but creates some further problems. As an example, a permanent Logic unit (Day) may never forward-reference a non-existent Day. This implies

that one cannot SAVE a Logic unit unless all Days to which it refers are already saved (and also that any Day referenced by another Logic unit may not be deleted). It seems most natural to proceed by saving the Days in the order in which a student would encounter them. This is not currently permitted because it would mean that (for a short time) a Logic unit would have a forward reference to a day which did not exist. The Logic units must be saved in reverse order; last one first, first one last.

This problem could be partially overcome by permitting the user to save a number of units simultaneously; but perhaps the best sclution would allow the user to "queue" a unit for saving. If he asked for a unit to be saved and it contained a forward-reference to a non-existent day (or was "caught" by some other restriction) it would be placed in a queue. As the user continued his work; either the restriction would be removed, in which case the unit would be saved automatically and a notification produced; or it would not, in which case the queued unit would simply be lost when the user logged off. Modifications to CATALOG (to print the queue) and PLACE (to move units from the queue into a workspace) would be necessary to complete this facility:

3. Another high priority change in the Mode Two command set is the addition of a RENAME command. Because of the restrictions discussed in point two above, changing the name of a unit had to be done in a slightly complicated but

definite way. One of the teachers who used Mode Two commented on the awkward procedure required. A duplicate of the unit had to be saved under the new name; then every Logic unit referencing that unit had to be placed in a workspace, manually changed, and then saved before the unit with the old name could be dropped. In addition to its awkwardness, this procedure also introduced opportunities for human error and caused a great deal of unnecessary movement of data. While changing the name of a unit is not as simple as changing the contents of a single location where the name is kept, the task could be accomplished internally much more economically than as described above.

4. The flexibility of the unit labelling allowed by TAIM can be both an advantage and a disadvantage. The disadvantage arises if the labels chosen are not meaningful to someone unfamiliar with the contents of the files. If TAIM were put into use with even a single course over two months or more, the number of units required would make a labelling convention necessary. While some of the commands could be made more powerful by having them "know" the labelling conventions, this could also make the whole system less flexible. The possibility of various degrees of fixed and unfixed labels for units must also be considered, and the solution based upon a trade-off between the amount of information contained in the label and the flexibility allowed. Such an analysis is outside of the scope of this study, but it seems clear that either the labels should be

made to contain more fixed information than that demanded by the implemented version of the TAIM systems, or a well structured convention should be established (at the outset of a materials construction project), documented, and made available to all users of the materials. Such a labelling convention or "context" is discussed in connection with the Conwell Approach in Chapter Two. A convention designed for use with TAIM is discussed later in this chapter.

5. A verb EXECUTE was planned for inclusion in Mode Two but not implemented because of doubts as to its usefulness. It would have enabled the Mode Two user to simulate DRIVER execution of the statements in a Logic unit, requesting information when necessary for evaluation of conditions, and printing labels of Displays or Tests to be produced. EXECUTEing a Test unit would have allowed the user to enter a set of responses and obtain a score on a particular test to validate the test key. While it seems unlikely that a user would discover an error either in Logic or in a text key that could not be discovered otherwise, the command would make such error checking more "fun" and more reliable. Being able to execute a Logic unit rather than waiting for an offline run of the DRIVER would provide immediate reinforcement. Such a facility might be more necessary if a larger number of Logic units were available, and/or if the Logic units were more complex than those used experimentally.

6. While the printout produced by the CATALOG command

was manageable for the experiment; as the number of permanent units became larger, the need for some means to obtain a more selective printout would increase. This could be based on a chosen labelling system (see point 2 above) but a more general solution that would be easily implemented would allow the user to specify two alphanumeric strings. The labels are printed in alphabetic order, and only those at or after the first string and at or before the last string could be printed.

7. Some of the Mode Two users felt that certain command names should be changed. Particularly, they felt that QUIT should have been STOP to correspond with STOPing, the EDIT and SETKEY sub-modes. While the designers of TAIM felt that it was more consistent to keep a distinction between STOP (return to TAIM) and QUIT (return to the resident system), changing the actual keyword is simply done as discussed in Chapter Three.

8. Several of the teachers felt that the DROP command should request a verification from the user before actually dropping the unit. This convention is used by MTS when destroying and emptying files. Some also felt that when a command was not executed, the error message should automatically be produced. These preferences are seen as variable across different users. The solution proposed is to add a new All Mode verb SET which would allow the user to set various global switches. (Such as requesting verification with DROP, or always printing the error message

if a non-zero return code is obtained from execution of a command.) The User File is updated each time a user signs off so that these switches could be kept and re-initialized at the start of each session. Possible futher uses for these switches are as follows: a) cause the user, when first starting a session to enter a mode other than the highest to which he is authorized; b) cause an exit from the system immediately after issuing QUIT; c) request confirmations for UNHOLD, UNNOTE, and UNREGISTER commands; d) enable or disable certain commands. Some of these switches might be alterable only by a Mode Three user; others by the user himself.

THE WORKSPACES

Three workspaces were provided by the system for communciation between the user and the permanent files. Some users felt this number to be sufficient ("any more would be confusing"), but others felt it restrictive. No great difficulty appeared to be encountered by any of the teachers in learning how to use the workspaces or in distinguishing them from the permanent units or the Files. Since a similar concept is used by both BASIC and APL, some transfer was expected to occur.

When doing a lot of manipulation of units however, keeping track of the contents of each of the three was difficult. This probably was a result of the very uninformative names (ws1, ws2, and ws3) of the workspaces. One solution to this problem would be to allow reference to

a workspace by either the workspaces name or its current label. (Workspaces are labelled with the LABEL command, or implicitly by the PLACE command.) This might result in confusion if two workspaces had the same label; but could be countered either by notifying the user and requesting the workspace name, or by not permitting the same label on two or more workspaces. Unlabelled workspaces would have to be referred to by name. Directly affected by this change would be the commands EDIT, EVALUATE, SAVE and SETKEY. If the user were permitted to refer to the workspaces by their current labels, allowance for more workspaces would likely be desirable. This alteration to TAIM Mode Two is seen as a high-priority requirement.

THE MAIN FILES

The Mode Two users were not really aware of how their actions affected the main files. However, these files were explained and through discussions, two possible additions to the main files were suggested. One would be a file of multiple choice or integer-answer questions from which Tests could be assembled. The Tests might be constructed by the Mode Two user simply moving the questions from such a file to a Test unit under construction. Another possibility is that the Test units might contain "rules" for the selection of test questions rather than the questions *per se*. The tests would then be assembled dynamically for each student by the computer during the DRIVER run. Students being evaluated on the same topic would receive identical tests

under the first option, but parallel tests under the second. Computer systems for dynamic test construction are presently available (for example the CTSS; see Lippey, 1970).

A second extension, the addition of a RESOURCE ALLOCATION File, was also suggested. This file would contain information on the resources that were available for student or teacher use, and under control of statements in the Logic units dynamically allocate these resources. It is not clear which resources might best be managed, how they should be allocated, or even if such management is feasible. For example, one can visualize entries of the number of filmstrip projectors; when instruction was given to a student requiring one of these, the number available for a particular time interval would be decremented. An algorithm for determining who used which equipment at what time would have to be devised. The possibility of sharing equipment would have to be considered. If such an extension was available, it might conceivably manage school resources ranging from teachers and classrooms through books and filmstrips down to pencils and paper. Individual preferences would also have to be considered. This particular facility is seen by Salisbury (1971) as an integral part of any CMI system. However, the mechanics of implementing such a facility are far from being resolved.

THE LOGIC FILE:

The Logic File statements appeared to be easily learned by the teachers. With the aid of the automatic syntax checking features of TAIM, the teachers constructed all of the Logic units by themselves. While the units could have been more elaborate than they were, not a single error was made by the teachers in construction of the units. As described in Chapter Three, the "conditions" in the IF and WHEN statements were designed to allow decisions based on portions of tests by appending the numbers of the desired questions to the Test names. This was not implemented, and although the experimental version of the curriculum operated successfully without this facility, it is proposed as a necessary addition.

THE DISPLAY FILE:

Display file unit construction took a large portion of the teachers' time. Making the Displays required the teachers to master concepts of printer carriage control and means of obtaining special characters available on the offline printer but not present on the terminal keyboard. They were required to attend to the details of spacing and arranging the textual presentations on the page. While this task is a complicated one, in a larger scale implementation such data entry and layout work would be more economically done by someone hired specifically for this purpose. Assembly of the initial contents of the Displays and some subsequent editing would likely remain as part of the Mode

Two teachers' tasks however.

Programs designed specifically to permit special characters and to arrange textual material are currently available. Examples are IBM's TEXT 360 and UBC's Format (this thesis was produced using the latter). Providing such a facility with TAIM would improve Display and Test text formats, but would require the mastery of an additional array of conventions by teachers and aides using the systems. It would have been possible to use either of these peripherally with the implemented version of TAIM but this was not done.

THE TEST FILE:

The Test File is subject to the same comments as above since the textual portion of each Test was identical to a Display. Each Test unit also contained a test key. The key was constructed through the SETKEY command, and the Mode Two users were generally very happy with this facility. The reason for the exceptionally high regard the teachers had for the SETKEY sub-mode is not known, but may have been partly due to its ability to print out the results of teachers' efforts upon request.

LEARNING TAIM MODE TWO

Learning TAIM Mode Two took the teachers an average of three and one-half hours of class time and study and an average of two hours working at the terminal. A formal examination showed that they had learned eighty per cent of

the concepts considered necessary by the writer. It was clear that what they had learned was sufficient to enable them to proceed with the construction of all aspects of the materials.

The teachers suggested that a two-day inservice would be sufficient to teach other teachers how to use TAIM. In the writer's opinion, two half-day sessions followed one or two weeks later by a half-day question period would be most appropriate; provided that participants had facilities to use TAIM during the interval. The teachers concurred that the most useful technique for teaching TAIM (or any other computer language) is hands-on experience. They further felt that a resource person more familiar with the overall operation of the system should be present in each school, and that both self-study and reference manuals were needed. The resource person would require additional training which might be obtained through further inservice or through self-study (perhaps under TAIM itself). This point is discussed further in section three of this chapter.

Although a number of suggestions for improvement to TAIM Mode Two are suggested above, the implemented version exceeded expectations in the facilitation of construction of instructional sequences and materials. The group of teachers spanned a range from little to extensive experience with computers and had similarly broad ranges of teaching experience and general motivation. While the quality of the materials they constructed varied considerably, each member

had the ability to generate the material properly in the system.

B) CONSTRUCTION OF MATERIALS

THE FLOWCHARTS

The flowcharts detailing the sequence of instruction for the specified period were considered essential for two reasons. Initially it was used to lay-out the total sequence of instruction sc that each unit cculd be constructed with appreciation of its role in the whole scheme. Subsequently it enabled someone not familiar with the contents of the files to obtain a grasp of the overall plan.

The use of a flowchart is closely related to the problem of how much information is contained in the unit labels. Anyone using a set cf prepared materials would want the facility to easily ascertain the position of a particular unit in he whole scheme knowing only the unit's label. Knowing the topic being covered, one would also want to be able to easily identify the relevant units by label. This information might be read from a flowchart, or might be implicitly contained within the labels themselves.

By way of comparison, consider the two problems of divining the contents of a book knowing only its title; and finding, by title alone, all books pertaining to a particular topic in a library. If everyone were to entitle their books according to the Dewey Decimal System (or some other) the problems would be simplified; but there are obvious reasons why this is not practical. These same

reasons; loss of flexibility, difficulty of standardization, possibility of excluded categories, aesthetic loss, and problems of multiple membership and universality; apply to the systematizing and encoding of unit labels as well.

Again, a trade-off or compromise solution is indicated. The TAIM system as presently designed leaves the solution to this problem completely in the hands of those who construct and maintain the instructional sequences and materials. For the experiment, no form of label organization was employed. Near completion of materials construction, the teachers became aware of the problem this had created and were annoyed with themselves for not forseeing it and with the writer for not insisting on a labelling scheme. In both the Preparation and Implementation procedures this forced a heavy dependence upon the flowchart and perhaps caused its value to be slightly over-rated by the participants.

PRINTER RESTRICTIONS

The printer restrictions offered some difficulties in the construction of Display and Tests. One problem that has already been mentioned was the availability on the printer of characters not on the terminal used for text entry. The most important such characters were the integer superscripts. A convention was arranged so that these could be obtained by using unique pairs of terminal characters. However there remained a number of desirable symbols not available on either the printer or terminal. A symbol for approximately equal was needed throughout the materials

discussing logarithms. Experiments with overstruck symbols were conducted tc find a suitable representation, but finally the teachers resorted to writing the words out in full or using "approx.=". It is possible to obtain printer chains which contain a great variety of characters. If TAIM were to be used sufficiently to warrant the facility, a chain containing all required characters could likely be obtained.

A second related prcblem is that of obtaining diagrams or figures. Constructing these with available printer facilities is usually difficult and the results are often unsatisfactory. Some of the problem could be overcome through the use of a special graphic printer which prints tiny dots instead of characters; but the degree of difficulty of "programming" diagrams, and the additional cost might make this solution impractical. Diagrams are clearly necessary for many teaching applications. The state of computer printing is currently such that conventional means of diagram production and reproduction are likely to remain most viable for some time to come. When using TAIM, the best solution appears to be to have all types of figures and diagrams prcduced by conventional means and then filed. The computer produced presentations can then refer to these as necessary. One of the teachers (Pf) suggested that special books could be prepared specifically for use with TAIM. They would ccnsist almost entirely of the necessary pictorial aids. The primary purpose of the TAIM printouts is

to help manage the students' learning. This does not demand the printing of all instructional materials by computer. Rather, materials should be produced in this way only when it is the superior alternative.

RESPONSIBILITY

Responsibility for the construction of materials is another problem that must be considered in terms of a trade-off or compromise. Without the provision of extra time, it would be difficult for a regular classroom teacher to produce an adequate array of Display and Test units for his classes. Further, it would be uneconomical for each teacher to produce his own array since this would involve a considerable duplication of effort. The only apparent feasible solution is to provide a group of teachers whose main task would be the initial construction of an extensive array of such materials. The problem of communicating what was available to the teachers who would eventually use the units would have to be resolved (perhaps arbitrarily). Suggestions made previously; use of flowcharts, increasing the information content of labels, and provision of various cataloging and cross-referencing manuals; would have to be considered.

Once such an initial array was constructed however, a division of duties is envisioned. For reasons discussed in Chapter Two, it is the classroom teacher who has the best information, the initiative, and the need to update or revise and add to the supplied units. A division of units

into PUBLIC (those constructed and maintained by the initial group) and PRIVATE (those constructed and maintained by individual teachers) seems desirable. All teachers would have access to the Public files, and would (by default) use the instructional sequences and materials there for their course. If they wished to make an alteration they would simply add a number of Private units which incorporated their changes. Whenever the system referenced a unit available in both the Public and Private files, the Private unit would take precedence. Systems of Public and Private files are currently in use at most computer installations.

A group of teachers in charge of the Public files would continue to make their own additions and alterations. They could transmit notification of their work to the classroom teachers through messages printed out with each teacher's DRIVER run. The group could be responsible for monitoring all of the Private files that the teachers created; making many of them Public files if they constitued additions to the system, or using them to replace Public files if they contained improvements. Under certain conditions, the "best" replacement for certain units could be determined by statistical means; a possible criteria being higher achievement on subsequent tests.

<u>TEACHER REACTION</u>

Teacher reaction to the required method of constructing instructional sequences and materials under TAIM could be classified as "mixed." However, in interpreting reactions, one must keep in mind the small sample (six teachers) who participated. The Attitude questionnaire seemed to indicate that while teachers became less "afraid" of computers (even Pf who had extensive prior experience with computers showed this tendency), they also shifted from a generally high endorsement to a more moderate endorsement of the use of computers for instruction. In retrospect, it appears that some of this shift might be attributed to the teachers simply having a more realistic concept of what computers can and cannot do as participants in the instructional process. This hypothesis was supported by the teacher's written comments and by personal communication.

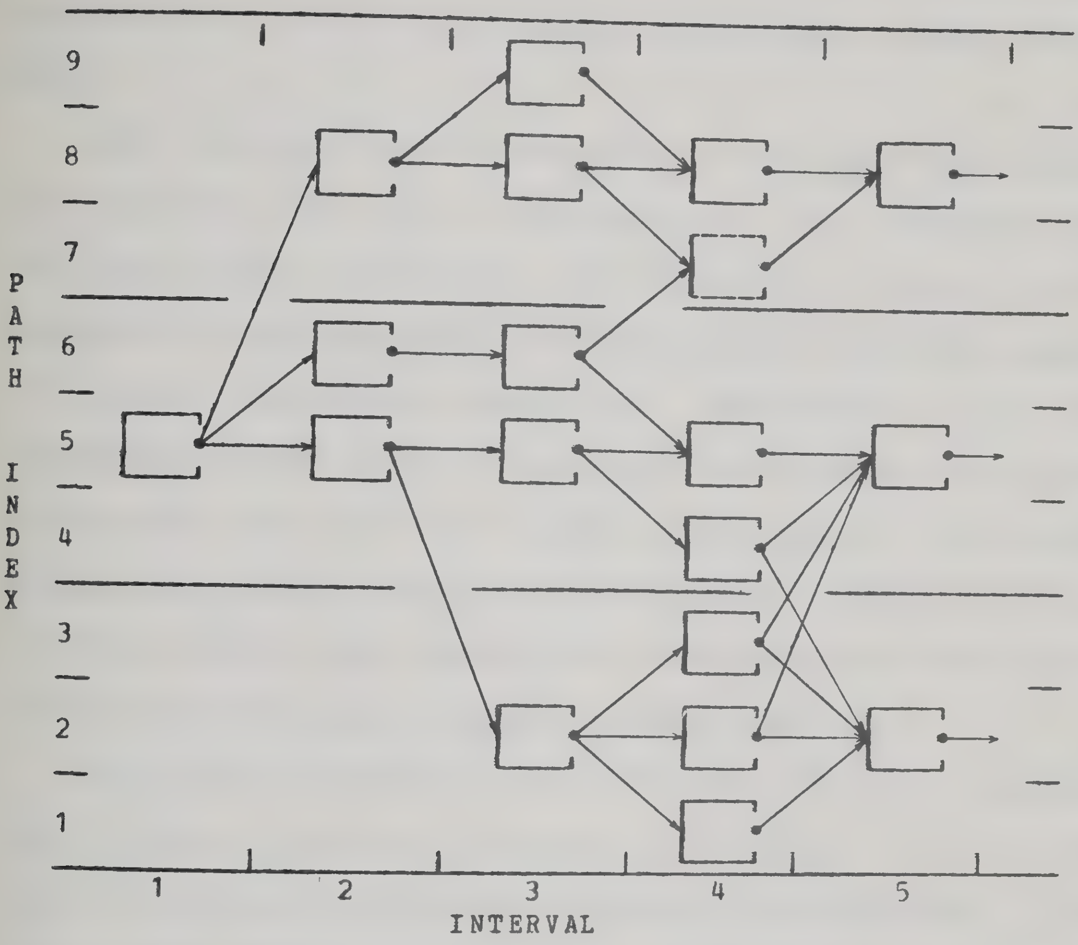
In using TAIM the teachers became increasingly aware of the need to attend to minute details in the planning of curriculum and in the construction of instructional sequences and materials. They were faced with the almost overwhelming complexity of attempting to plan in advance appropriate activities for learning for disparate pupils; complexity which the classroom teacher rarely encounters in the course of traditional teaching. To the writer and to the teachers themselves, the ability to cope with this complexity seemed closely related to the teacher's ability to teach. Those who did not have prior teaching experience

cited this as reason for difficulties they were having; those who did have experience were aware that it was a valuable asset. While learning how to use TAIM is a relatively simple task, the ability to construct appropriate instructional sequences and materials requires a high level of professional pedagogical expertise.

C) <u>A LABELLING CONTEXT</u>

As discussed previously, there is a need to maximize information on the contents of a unit in the unit's label. Both the amount and type of information in the label may vary. The Conwell Approach description in Chapter Two contains an explanation of their method of item labelling. Each label was a fourteen digit number and contained codes indicating such things as content area, reading level, difficulty level, etc. They did not contain any sequencing information (ie: the order in which units may be distributed) but then the Conwell System operates under a "search" hypothesis; sequencing being determined externally. TAIM, in contrast, manages both sequencing and searching but requires that the instruction authors specify the rules for making the decision at each decision point.

During construction of the materials and during the Implementation procedure, it was confirmed that some guiding structure for the units was necessary. Even with the few Logic, Display and Test units involved in the experiment, it was often necessary to refer to the flowchart for reference purposes.



LABELLING CONTEXT CHART

FIGURE 13.

Central to the labelling context discussed here is a flowchart imposed on a grid structure. Because of the importance of the capability provided by TAIM for sequencing student progress, this information is prominent in the chart. Each box in the chart represents a Logic unit; the lines joining the boxes depict the logical sequencing of the units. The flowchart proceeds from left to right so that the horizontal axis of the underlying grid could be labelled "Time." However, since most courses would be designed to

proceed sequentially through various items of content, this axis may also indicate the topics contained in the Logic units.

The vertical axis of the grid is labelled "Path Index" and divided into nine categories. It is possible to visualize these categories distinguishing reading level, difficulty level, aptitude, learning style, cognitive style, or another dimension which the constructors of the curriculum might wish to emphasize. The chart is restricted to only two axes, but the vertical could permit a hierarchy of some type. For example, if three reading levels and three learning styles were identified, the categories would allow positions for the nine possible combinations.

The intent here is for the vertical axis to represent three student ability groupings; categories 1,2, and 3 the lower; categories 4, 5, and 6 average; and categories 7, 8 and 9 the higher ability stream. While such a division may not seem very scientific, it has the advantages that it does not rule out further classification on the variables already mentioned, and it is the writer's observations that teachers frequently tend to use this type of classification as a part of their instructional planning.

Each of the three streams is conceived to contain a central instructional path (indices 2, 5, and 8), an enrichment path (indices 3, 6, and 9), and a remedial path (indices 1, 4, and 7). An average student would normally receive lessons controlled by the units depicted along path

5. At intervals, he could also receive enrichment (path 6) and remedial (path 4) work appropriate to his ability and current needs, and possibly as determined by measured variation on the variables mentioned previously. A bright student, on the other hand, might meet or surpass a determined criterion enabling him to enter the top stream for a particular topic. He would also have the possibility of receiving enrichment (path 9), or remedial (path 7) lessons geared to the type of presentation being used and to the level of instruction. There would not necessarily be any common material or even any similarity between, say, average enrichment (path 6) and top stream remedial (path 7) work. Any information which cannot be implied by a positioning on the grid but which is nevertheless required should be displayed inside the box.

Many of the units represented in the chart would have more than one possible exit, leading to alternative sequences of instruction. Where this occurred, a decision must be made and it is likely that the basis for this decision would be an important candidate for inclusion in the box. Other items which might be so displayed include the label of the Logic unit, the labels of Displays and Tests referenced, an indication of the specific subject to be discussed, lists cf materials or facilities that might be required, messages that might be produced, and the objectives intended to be met.

A main disadvantage of the labelling context is that it

does not easily permit depiction of non-linear sequences. For example, it would be difficult to show the possibility of a branch from a unit in interval 10 to a unit in interval 5 or 20; yet this is easily done with TAIM logic. This makes it difficult to indicate "looping"; where a student might repeat a sequence of units until he met some criterion.

A second major disadvantage is that through the use of IF statements, an individual Logic unit can be made to generate several completely different lessons. The only means of indicating this on the chart would be to include all possible outcomes in each box. The context presented here would thus discourage the curriculum builders from using this facility.

Once this or any such labelling context was established, the labels of all units constructed would incorporate aspects of the context. For example, the Logic unit in interval 3, path 9 might be labelled L-3.9:INV-FNS:AV to supply its position on the grid, indicate the subject (inverse functions), and warn that some A.V. equipment might be required. An associated Display and Test could have the same label; if more than one Display were used, sequence numbers or letters could distinguish them.

As mentioned previously, this particular context would not be ideal for all applications. The context chosen depends upon a number of factors, not the least of which are the desires and intentions of the initial curriculum system

planners. They are the ones who must decide; on the basis of the students to be taught, the subject to be presented, and their ability to provide alternatives; which among the possible variations will be used.

It may develop; as more research is done into learning, teaching, and the role of the computer in education; that a specific context (and implicitly a means for organizing instruction) can be found which best meets all needs. But more likely, such organizations will vary between applications. TAIM provides the facility for such variation and is therefore a good vehicle for experimentation in this area.

2. IMPLEMENTATION FEASIBILITY

The Implementation procedure consisted of using the instructional sequences and materials prepared by the Ed.CI. 466 class with a Math 20 class and then with two Math 33 classes. Two teachers were involved, and they used the Mode One portion of TAIM and the DRIVER program to produce the instruction for their students.

A) MODE ONE

THE MODE ONE COMMAND SET

The Mode One command set, taken as a whole was only adequate in allowing the teachers to control the instruction produced by the DRIVER and to obtain information about the students. However, a few relatively minor modifications would upgrade the set substantially.

[illegible]

1. The READ command could use two main improvements. Because it was initially planned to be used with a peripheral optical mark card reader it accepted student responses of 1, 2, 3, ... Teachers and students are accustomed to using 'a', 'b', 'c', ... for responses to multiple choice questions, but this attribute of READ forced them tc convert. While the potential for integer value answers should be maintained, READ could be modified to translate 'a' to 1, 'b' to 2, etc., so that either integers or letters would be accepted.

As soon as the teachers had entered all student responses, they then wanted to know the results. That is, what marks the students had received and to what Logic labels they had been pointed. This information is actually generated by READ, and is known as soon as each student's response card is read. A minor modification would cause READ to print this information as it became available. This could be controlled by an optional parameter.

At the terminal, the proximity of entering a student's identification code and responses and seeing his results would help the teacher to "keep in touch" with individual students in a way similar to when teachers mark student exams by hand. This proximity would also help to reduce errors in transfer of responses from cards to the terminal and enable teachers to test the effects of certain patterns of responses. If a card reader were available, translation errors would be minimized, and the information would be

printed on the terminal while the cards were being read.

2. A second major problem encountered by the teachers was that the current version of TAIM has no facility for directly determining the students for whom cards were not read. This information is indirectly determinable; both MARK and TRACE provide it when they are issued across all students. Because of the implementation of the READ command, teachers always issued either or both of these and thus were not greatly inconvenienced by lack of this facility. However, change to the READ or a large increase in the number of students would make this a more pressing problem. The change suggested is to allow recognition of a special ITD-label "*POINTED*" (or just '*') on the WHO command. Users could then ask "who class-a missed *pointed*" (or "who class-a m *") to obtain a list of all students whose responses had not been entered. ("who class-a *" would list the IDs of students who had been pointed.)

It was necessary to identify students whose responses had not been read. The DRIVER interprets a missing response card to mean that the student was absent and therefore had not learned the previous material. It reproduces the material for the student again (and again if necessary) until the responses are supplied or the teacher intervenes. In the Implementation procedure, the teachers did not want students to get "out-of-step" and so they usually POINTed students who were absent to the next sequential lesson. In some cases they used the NOTE command to include the

"missed" material in the new lesson as well. This problem was aggravated by two factors: because the experiment was conducted at a time near the mid-term break, absences were high; and since the experiment lasted only eight school days, everyone had to be kept together so that they would all finish at the same time.

Nonetheless, the default action of the DRIVER seems appropriate. Near the end of the experiment the teachers discovered a way to circumvent this default action. They would POINT the whole class to a chosen label before reading in the response cards. The READ, occuring after the POINT, would replace the default label with one determined by the logic. The next-lesson pointers of the students for whom responses were not supplied would not be changed and remain at the default setting chosen by the teachers.

3. A third major problem arose about four times during the experiment. Responses would be available for a student but TAIM would not accept them because it had not supplied the student with the test. This happened in two ways. In one case, a student joined the course after the experiment had begun. He was not registered to TAIM, but was given the printout of a student who was absent that day. After class the teacher registered this student to TAIM, but was unable to have the system accept his responses. In another case, a student was absent for one day and the teacher pointed him ahead. Subsequently, he provided responses for both the test he had missed and for the current test. TAIM would accept

the latter responses, but would not permit entry of those for the previous test.

Marking the student's tests and placing him on the appropriate instructional path was easily done. The only difficulty that was not resolved was having the record of these actions placed in the student's file. The Student File is not directly alterable by any MONITOR commands. It is indirectly altered by the NOTE and POINT commands; the DRIVER program and the READ command control the opening and closing (respectively) of individual student's records. Providing the teachers with the capability of altering these records to make it <u>seem</u> like a student took a test on a day he was absent or before he was registered does not appear to be desirable. These records are meant to provide a trace of what actually occurred. However, if a student has studied a particular lesson and answered one or more tests, there is good reason to have this so noted.

A new Mode One verb "<u>IN</u>CLUDE sids LTD-label" is recommended. In the cases mentioned previously, the teacher would, before reading in responses (and closing the record) of the particular student, indicate that they had also been given particular Displays and Tests. These would be entered as if they had actually been produced by the DRIVER, but in the record pertaining to the time at which the students actually received them. The student who missed a day and then covered two days of material when he returned would have this recorded. The record for the day he missed would

show him as absent, but the next record would contain results for both tests he had taken and note the Displays he had seen (if the teacher specified them).

4. Two further changes are suggested for the Mode One command set. Presently, there is no facility for the calculation of statistics on test results across classes. Whether such a facility is necessary in the individualized instructional setting possible under TAIM is unclear, but this was requested by the teachers and would involve only minor programming. This should be provided by default with both the AVERAGE and MARK commands; the commands could also permit suppression of all other information. Statistics suggested are group mean, standard deviation, and the highest and lowest scores.

The second suggestion is to add a fifth option "LABELS" to the DISPLAY command. This would cause the output of all labels currently being referenced, and indicate for the Logic units whether they were "active" or were referenced by next-lesson pointers (or both), and for the Test units whether the test had been given or the scores were being used for lesson determination. This provision would help remove some of the confusion discussed previously under the labelling context, and enhance the power of the WHO command.

B) TAIM IN THE CLASSROOM

CLASSROOM PROCEDURE

The classroom procedure used with TAIM is subject to a great deal of variation. In the Math 20 and one of the Math 33 classes students were asked to sit in a certain order. The printouts, which were arranged alphabetically by student identification code, were then handed to one of the students who distributed them to the class. In the other Math 33 class, the teacher stood in the center of the room and called out the students' names.

The class teacher then usually gave a five minute verbal introduction to the lesson and the students began reading their printouts and doing the described activities. The teacher (and the writer and any visitors if present) then moved from student to student, helping out where necessary. Students freely talked to each other, and several loose groups formed.

The class periods were eighty minutes in length and with about twenty minutes remaining the response cards were distributed. Students were responsible for turning in these cards before they left the class or arranging to get them to the teacher later in the day. The teacher (or a student) then entered the responses from the cards at the terminal in the school. That evening another DRIVER run was initiated and the lessons were delivered to the school the next class day.

One of the Math 33 classes met in the morning, the

other in the afternoon. The writer deliberately did nct appear at the afterncon class until the last day; visitors were discouraged, and none attended. On the fourth day with this class, the teacher felt he had misled the students in his introduction. Subsequently he gave no more introductions and put his trust in the system and the materials.

Whether this was an acceptable thing to do is debatable. Certainly, if TAIM were to be used widely, a large number of teachers might be expected to take this action. Not being familiar with the overall plan of the instruction, they might feel that talking to the whole class would interfere with the planned process. They might fear disrupting the sequence and even believe that everything was so well planned that anything they did could not improve but could only perturb the students' learning.

At the current state of knowledge of instruction and the learning process however, there is reason to believe that the classroom teachers will be better able to provide instruction than any CMI system. As argued in Chapter Two, it is the teacher's perception of the process that is expected to help improve the computer-supplied instruction. Placing too much faith in the system destroys this necessary feedback loop.

With the Math 20s, on about five of the eight days, a significant proportion of the students finished their lessons with a substantial amount of class time remaining, or had to stay after class to complete their work. This is a

perennial problem with individual lessons because of the great disparity in the rate at which students work (or a student works). With TAIM, it is possible to vary the amount of work given to individual students, but estimating, *a priori*, how long a particular task will take is difficult. In preparation for the Math 33 classes, adjustments were made. Some Displays were moved from "long" days to "shorter" ones, and some Displays and Tests were altered. The ease of making such changes is a major advantage of TAIM; even with only one trial with a different grade level class, enough experience was gained to make the adjustments quite accurate. The problem of large numbers of idle or "rushed" students was largely overcome for the Math 33 classes.

While large groups of students did not finish too quickly or too slowly, individuals in the Math 33 classes were forced to hurry or left with little to do. This simply indicates that finer adjustments are necessary. However, the tendency is to give students who obtain high scores more work and those with low scores less. Whether achievement scores are appropriate criteria for determining the amount of work assigned is open to question. TAIM allows sufficient control of instruction to permit experimentation in this area. Perhaps a better answer can be found.

STUDENT ACHIEVEMENT

Student achievement was assessed as discussed in Chapter Five. Table 7 shows that the Math 33 students' total final scores did not differ markedly from their pre-test scores. As was mentioned, the pre-test was very short (see Appendix C, item 5) and was designed not to discriminate on levels of knowledge but to determine whether students knew anything about logarithms at all. While the Math 20 class had not taken logarithms before, many of the Math 33a and nearly all of the Math 33b students had recently studied this subject. These students obtained higher scores on the pre-test, which contributed to the low mean difference reported in a comparison with the total final exam (see Table 7). Comparing the scores for all classes with a parallel test selected from the final did show (see Table 8) a marked improvement however.

STUDENT COMMENTS

Student comments on the system showed that they had a fairly accurate perception of how TAIM worked and that they liked receiving their instruction in this way. Their remarks, in Chapter Five, are to a large extent self-explanatory. It is significant that about one-quarter of the students were unable to supply one thing that they disliked about TAIM to a direct question. Their remarks about "being rushed" and "full attendance" were a natural reaction to the teachers' decision to keep everyone moving at the same pace.

<u>TEACHER REACTION</u>

Teacher reaction to the use of TAIM with their classes was quite positive, but this was based mostly on what they perceived as potenial use of the system. Whether they would be enthusiastic after prolonged use and even dependence upon TAIM is a question not answerable by this study. The teachers did not actually make any alterations to the materials themselves, since this required the use of Mode Two. Rather, changes were discussed and then implemented by the writer. However they were keenly interested in the procedures involved.

No formal instruction in the use of the Mode One command set was given; yet the teachers were about as familiar with it as the writer after a few days. They regretted the short duration of the experiment, and commented that they were just beginning to make full use of the NOTE command when the trial ended. They learned the use of the system by watching the writer, by reading a manual, and by trial and error. (TAIM has an extensive error handling facility; as one of the teachers commented, it is possible to learn how the system works by just typing things in and reading the error messages.)

The results of the Implementaion procedure showed no reason why TAIM could not be put into more widespread use. However the conclusion that it is currently feasible is not obtained because of the small numbers of participants and, more importantly, the short duration of the experiment. The

teachers involved were specifically chosen because the writer felt they would not be antithetical to TAIM. Whether a majority of teachers would welcome the system is not clear. It is likely that at least some teachers would not be able to proceed to use TAIM in an acceptable way.

The classroom teachers were involved with TAIM for less than four months. Actual use of the system lasted only sixteen days, spanning an interval of just over one month. One of the teachers was head of the school's Mathematics Department, and was responsible for teaching only half-days. He therefore had more time to devote to TAIM than an average teacher. For both teachers, the subject taught by TAIM was one they were most capable of teaching themselves; and the TAIM-taught classes represented only one-quarter of their total duties for the interval of the experiment. While the Implementation procedure did not show that this part of TAIM was not feasible, the special circumstances above prevent any more firm conclusions from being reached. However, the degree of acceptance of the system, even over the short span, would indicate that more testing is desirable and even demanded.

3. <u>COST FEASIBILITY</u>

With a system as complex and as flexible as TAIM, it is difficult both to collect meaningful cost data, and subsequently to summarize it in such a way as to be useful to others. TAIM was implemented on the University of

Alberta's IBM /360-67 via the PL/1 programming language. The costs reported here are those assessed by the Department of Computing Services at 100% rates; the effects of PRIORITY FACTOR (see chart in Appendix E) were removed for offline processing, but kept for online processing making the figures reported the maximums that could be assessed. Costs were based on the charging algorithms used during the fiscal year 1972-73, and are already out of date; most resources are now slightly less expensive. The fact that the computer language PL/1 was used is significant. This language was chosen primarily to help reduce the length of time it would take to program TAIM. However, because of the nature of PL/1 and partly because of the implemented version at this University, the final result is more costly to run than if a language such as FORTRAN had been used.

In some sense, it may be true that "Any fairly good programmer can set up a computer-managed instructional system (Gibb, 1973);" but the relative costs of such systems cannot be compared without a simultaneous assessment of the quality and appropriatness of the system's functions.

PREPARATION

Preparation for the use of TAIM with students involved the training of a number of people to use the system, specifically to use Mode Two. Results from this experiment suggest that this could be accomplished with approximately six to eight hours of instruction, preferably spread over an interval of at least a week, combined with laboratory

activities and unstructured hands-on use of the system during the interval. The laboratory activities should be arranged to provide as broad an experience as possible and should be appropriate to the intended eventual use of the system. The procedure of simultaneously beginning the design of the overall plan of the instructional sequences and material was satisfactory.

Contrary to the comments made by the teachers, initial planning for instruction is seen as a major task. However teachers already familiar with the contents and desired sequence of a particular topic, may be able to proceed quite quickly. Since it is very easy to make alterations to units, actual production of materials can occur parallel to instructional design, and as the overall plan becomes more established, materials can be rearranged and modified to fit where needed.

Where the volume of text to be entered was relatively large, input would best be handled by someone adept at typing. This cost is again subject to factors outside the scope of this study; but approximately $1 per double spaced typewritten page would seem adequate for the text of Displays and Tests. Construction of Logic units and test keys might best be done by teachers, because the task of deciding what is to be input is considerably more time consuming than the typing itself.

The cost of using Mode Two for the input and arrangement of the curriculum depends chiefly upon the

efficiency of the user. In absolute terms, the amount may seem considerable. A total cf $130 was used in Mode Two for the six teachers to construct the required materials and for all the revisions done. Fifty-eight dollars were used for SAVEing units, $28 for EDITing and $16 for PLACEing permanent units in workspaces. The costs would have been higher if all of the input had been done under TAIM. However, comparing these figures to what would have been the cost if the personnel had been hired to prepare and revise the material puts the computer charges in proper perspective; they are insignificant.

IMPLEMENTATION

Implementation of the prepared curriculum in a regular class using TAIM incurs charges which are significant however. The two main sources of computer cost here are the running of the Mode One portion of the MONITOR, and running of the DRIVER. The cost of using the MONITOR depends directly upon what type and how many commands are issued. A third important factor is whether the commands request output immediately at the terminal, or defer processing for the overnight run.

During the Implementation procedure, a total of 531 Mode One commands were issued online for a total cost of $77. Twenty-five dollars was used for READing student responses, $17 for MARK retrieval, and $7 for TRACEs. Sixty-one Mode One commands were issued for offline processing with a total cost of $18. Of this $9 was spent on MARK, $6

on AVERAGE, and $3 on TRACE commands. This gives an overall total of about $4 per class-day spent on MONITOR use. However this figure is severely inflated because the system was being used in an experimental situation, the costs of the teachers learning Mode One is included, and some initial costs (such as registering all students - $10) should be amortized over a longer period. The present system, in a production environment, would probably run for about $1 per class per class-day (where "class" is a group of from 30 to 100 students). This could be reduced to about $.75 if teachers consistently queued their commands for offline processing.

The cost of running the DRIVER program varies with two main factors: the number of students registered, and the amount of processing done for each. Analysis of the data of Table 11 shows that the TOTAL cost of processing for a student is closely related to the number of lines printed for him. Graphs of cost versus number of lines for the SHOW, TEST, NOTED, and TOTAL are all similar to the graph of the function COST=LINES/7 + 2. Defining a "page" as 28 lines of textual material, the resulting equation indicates that the cost of producing one student's printout can be determined as a basic two cents plus four cents per page. The lines on this page may be up to 132 characters in length. The four cents is attributable to the transferring of the text to a file for later printing plus the actual cost of printing. The basic two cents accounts for all other processing such

as "recognizing" the student, determining his pointers, and intrepreting the necessary Logic statements.

Using this formula, one can arrive at a fairly accurate predicition of the cost of producing and printing the students' lessons. This value does not include costs attributable to other necessary process in the overall offline run (such as sorting the printout, maintaining backup, and overhead) as discussed in Chapter Five and displayed in Table 16. Examination of this data shows that the total cost of the DRIVER run (less the cost of the MONITOR) can be estimated at five dollars plus seven fourths of the predicted cost of running the DRIVER program.

These two results may then be combined to produce the formula

$$T = \frac{7N(1 + 2P)}{2} + 500$$

which yields T (the predicted total cost of the DRIVER run in cents) from N (the number of students), and P (the average number of pages to be printed for each). For the Math 20 and 33 classes, the average number of pages printed for each student was six. The predicted average cost of the DRIVER runs (less the cost of running the MONITOR) for the Math 20s (n=37) is $21.84; the actual cost was $22.04. For the Math 33s (n=57); predicted is $29.13, actual was $29.27.

SUMMARY

In summary then, running the offline programs (excluding the MONITOR) appears to incur a basic charge of five dollars plus three and one half cents for each student registered plus seven cents for each page of student printout produced. Running the MONITOR was estimated at one dollar per class day.

Extrapolation of this data is somewhat dangerous, but results of interest are obtained. A class of thirty students, receiving a printout averaging four pages a day during a 180 class-day year would incur a total cost (DRIVER plus MONITOR) of $15 per day or $2700 for the year. This amounts to about $90 per student per year or fifty cents per student per day. This is approximately equal to the cost of providing the teacher for the same class.

As a second example, consider an undergraduate University class of 20 students receiving a printout (4 pages) each week for a total of 25 weeks. The total cost is predicted at under $12 per week or just $400 for the year.

As a final example, consider a medium sized high school of 30 teachers and 1000 students. Each student receives an individualized printout of about ten pages directing his activities for all of his classes for the day. The computer bill would amount to roughly $750; but if students averaged between 7 and 8 classes each, the cost per student per class is about ten cents.

The equation as derived from the data on this

experimental system predict a minimum limit of seven cents per page of student printout. This limit is approached rapidly as the number of registered students increases, and as the number of pages in each student's printout increases.

D) RECOMMENDATIONS

The evidence generated by the experimental implementation of TAIM would indicate that this system is currently feasible for certain applications. It appears to be less costly to use than other individualization of instruction systems which involve the use of computers, but the cost is still too high for small-scale use in public schools. In an important sense, the system as implemented on the University's computer was not reliable enough. Businesses (including school boards) usually require that their computers do certain necessary tasks without fail (for example, payrolls). At the University however, the software is constantly being changed. During testing with the Math 20s, an alteration was made to the system sorting program causing the DRIVER run to fail. The nature of the error was such that all of the students had to be re-registered. This problem had been partly forseen and fortunately the student printouts were obtained only a half hour late. About one week after testing, a new version of the MTS System was introduced at the center, and neither the MONITOR or the DRIVER would run under it. The problem was located in the new Loader - something over which the writer had no control. It was repaired the next day, but if this had occurred

during testing students would not have received printouts for that class.

If reliability conditions could be met, there appear to be two conditions under which use of the system as presently constructed could be recommended. If relatively high cost was already a factor in the provision of instruction (as is the case in Universities, technical schools, and many retraining courses), TAIM could be used to effectively improve the instruction and perhaps even reduce costs. Secondly, if the degree of control and feedback provided by the system was necessary (as would be the case for certain experiments with curruculum or learning variables) or desirable (perhaps in business training courses), TAIM would provide a possible solution.

Larger scale implementation however, does appear to be feasible - even for grade schools. Many school boards have computers which they seldom use during the evening hours. The major cost of running TAIM is incurred during this time. Even at the University of Alberta, where total computer use is heavy, for most of the year during the evening hours the computer would be capable of doing more processing. While it is unlikely that TAIM would run entirely during "idle" time or even that school boards' present computers would be capable of running TAIM for a large number of classes, economies of scale and fuller utilization of resources would tend to make large scale use of the system less costly per unit of instruction.

By allowing teachers the capability of storing instructional materials and algorithms for instructional sequencing, and by providing the facility for ready modifications of these; TAIM provides a means for using information feedback to improve the instruction process. By automating the accumulation of large quantities of minute data on student differences and by providing the means for quick access to this data for use in determining instructional strategies, TAIM provides a practical means for the individualization of instruction. And by providing the means for retrieval of any of the information available and the facility for intervention by the teacher in any of its processes, the system remains under the control of those responsible for instruction.

SELECTED REFERENCES

Andrew, G.M., & Moir, R.E., Information Decision Systems in Education, Itasca, Illinois, F.E. Peacock Publishers, 1970.

Baker, F.B., "Computer-based Instructional Management Systems: A First Look," Review of Educational Research, 41;1, February, 1971.

Banghart, F.W., Educational Systems Analysis, Toronto, Macmillan, 1969.

Bell, D.A., Intelligent Machines: An Introduction to Cybernetics, London, Pitman and Sons, 1962.

Blanchard, G.F., & Cook, D.L., "Project Management and Educational Change," Educational Technology, October, 1971.

Bloom, B.S.(Ed.), Taxonomy of Educational Objectives - Handbook I: Cognitive Domain, New York, David McKay Co., Ltd., 1956.

Bloom, B.S., "Learning for Mastery," Evaluation Comment, Vol.1, No.2, May, 1968.

Bloom, B.S.(Ed.), Handbook on Formative and Summative Evaluation of Student Learning, New York, McGraw Hill, 1971.

Chorafas, D.N., Systems and Simulation, New York, Academy Press, 1965.

Clark, M.J., "The Neucleus of a Unified Curriculum," Educational Technology, November, 1971.

Coulson, J.E., Computer-Assisted Instruction Management, Mimeographed, System Development Corporation SP-3300, February, 1969.

Coulson, J.E., "Computer-Assisted Instruction Management for Teachers," AV Communications Review, Vol.19, No.2, Summer, 1971.

Crowder, N.A. & Martin, G.C., Adventures in Algebra, New York, Doubleday, 1960a.

Crowder, N.A., "Automatic Tutoring by Intrinsic Programming," in Lumsdaine and Glaser, 1960b, pp286-298.

Deutch, M.J., "The Applications of Cybernetics to the Professional and Scientific Operations of the U.S. Government," Third International Congress on Cybernetics, Namur, Association Internationale de Cybernetique, 1965.

Drumheller, S.J., "A Transitional Support System for the 1970's: Developing Individualized Instructional Programs," Educational Technology, October, 1971.

Eckman, D.P. & Mihajalo, D.M., "On Some Basic Concepts of the General Systems Theory," Third International Congress on Cybernetics, Namur, Association Internationale de Cybernetique, 1965.

Edwards, W., "Men and Computers," in Gagne,1962.

EMDA Committee, "Elementary Mathematics Developmental Activity," Mimeographed, Edmonton Public School Board, August, 1971.

Flagle, C.D. et. al., Operations Research and Systems Engineering, Baltimore, Johns Hopkins Press, 1960.

Flanagan, J.C., "Functional Education for the Seventies," Phi Delta Kappan, September, 1967.

Flanagan, J.C., "The Role of the Computer in PLAN," Journal of Educational Data Processing, Vol.7, No.1, February, 1970a.

Flanagan, J.C., "Individualizing Education," Education, Vol.90, No.3, February-March, 1970b.

Flanagan, J.C., "The PLAN System for Individualizing Education," Special Report, National Council on Measurement in Education, Vol.2, No.2, January, 1971a.

Flanagan, J.C., The PLAN Educational System: A Program for Learning in Accordance with Needs, mimeographed, American Institutes for Research, Palo Alto, California, July, 1971b.

Fogel, R.L., "An Approach to Program Evaluation," Educational Technology, November, 1971.

Gagne, R.M., Psychological Principles in Systems Development New York, Holt, Rinehart, and Winston, 1962.

Gibb, E.G., "The Computer - A Facilitator in Management and Instruction," The Mathematics Teacher, Vol.66, No.1, January, 1973.

Gagne, R.M., The Conditions of Learning, New York, Holt, Rinehart, and Winston, 1970.

Graham, E.M., "Individualized Instruction: Distinguishing Characteristics," The Arithmetic Teacher, January, 1972.

Hatfield, L.L., Computer Assisted Mathematics Program: First Course, Scott Foresman and Company, Glenview, Illinois, 1968.

Henderson, G.L., "Individualized Instruction: Sweet in Theory, Sour in Practice," The Arithmetic Teacher, January, 1972.

Hobson, E.N., Empirical Development of a Computer Managed Instruction System, Florida State University, Tech Report No. 8, 1970.

Hunt, D.E., Matching Models in Education, Toronto, OISE Monograph 10, 1971.

Isaac, J., "Computer Assisted Instruction Applied to the Teaching of Logarithms - An Evaluative and Descriptive Study," Unpublished Ph.D. Dissertation, University of Alberta, 1972.

Kaufman, B., Letter to the Editor, Educational Technology, November, 1972, p76.

te Kampe, B.G., "Individualized Instruction in Grade Seven Mathematics: the Teacher's Role," Unpublished M. Ed. Thesis, University of Alberta, 1970.

Knirk, F.G. & Gentry, C.G., "Applied Instructional Systems," Educational Technology, June, 1971.

Kumar, V.K., "The Structure of Human Memory and Some Educational Implications," Review of Educational Research, December, 1971.

Landers, R.R., "An Approach to Humanizing Education Through Technology." Educational Technology, June, 1971.

Lippey, G., Classroom Teacher Support System (CTSS), mimeographed, Los Angeles Unified School District, 1970.

Lumsdaine, A.A. & Glaser, R., Teaching Machines and Programmed Learning, Washington, D.C., N.E.A., 1960.

Mager, R.F., A Method For Preparing Auto-Instructional Programs, Palo Alto, California, Varian Associates, December, 1961.

Mager, R.F., Preparing Instructional Objectives, New York, Xerox Corp., 1962.

Martorella, P.H., "Curricular Change: A Paradigm for Analyzing the Parameters of Curricular Reform," Educational Technology, December, 1971.

Oettinger, A.G., Run Computer Run, Harvard University Press, Cambridge, Massachusetts, 1969.

Oliva,F., Maguire,T., & Pacey,R., IPI Project Report, Alberta Human Resources Research Council, 1972.

Popham, J.W., "Objectives and Instruction," Instructional Objectives, Chicage, AERA Monograph 3, 1969.

Pressey, S.L., "A Simple Apparatus which Gives Tests and Scores - and Teaches," School and Society, 23:373-376, 1926.

Remai, H.A., "An Experimental Investigation Comparing Attitudes Toward Mathematics of Modern and Traditional Mathematics Students at the Junior High School Level," Unpublished M.Ed. Thesis, University of Alberta, 1965.

Salisbury, A.B., "Computers and Education: Towards Agreement on Terminolcgy," Educational Technology, September, 1971.

Schein, E. H., Organizational Psychclogy, Prentice-Hall, Englewood Cliffs, N.J., 1970.

Simon, H.A., "The Architecture of Complexity," in Proceedings of the American Philosophical Society, Vol.106, No.6, December, 1962, pp 467-482.

Sunde, A.K., "Individualized Instruction in Grade Seven Mathematics: Pupil Achievement and Grouping Procedures," Unpublished M. Ed. Thesis, University of Alberta, 1970.

Skinner, B.F., "Teaching Machines," Science, Vol.128, No.3330, October, 1958.

Tuckman, B.W. & Edwards, J.K., "A Simple Model for Instructional Design and Management," Educational Technology, December, 1971.

Webster's Seventh New Collegiate Dictionary, Toronto, Thomas Allen & Sons, 1969.

Westrom, M., "Individualized Instruction in Grade Seven Mathematice: Rationale, Description, and Feasibility Report," Unpublished M.Ed. Thesis, University of Alberta, 1971.

Wiener, N., _The Human Use of Human Beings_, New York, Doubleday Anchor Books, 1956.

Winer, B.J., _Statistical Principles in Experimental Design_, New York, McGraw Hill, 1962.

Zarsky, D.E., "Design and Implementation of a Computer Based Teacher-Authored Instruction Manager (TAIM)," Unpublished M.Sc Thesis, University of Alberta, Fall, 1973.

APPENDIX A

COMMAND SUMMARY

This appendix contains three tables detailing the implemented Mode One, Mode Two, and Mode Three commands. Each command is given in syntactic notation as discussed in Chapter Three along with a brief description of its use.

APPENDIX A

MODE ONE

COMMAND SYNTAX	ACTION
AVERAGE sids [FULL]	Print averages across tests for students
DISPLAY sids {HOLDS|MESSAGES|NOTES|REGISTRY}	Print requested information
HOLD sids #days	Suspend sids printout for #days class days
MARK sids [FULL] LT-label	Locate & print sids marks on specified test
MODE {2 | 3 | ?}	Change to new mode or ask current mode
NOTE sids [[D-]label]	Queue additional message or Display for sids
POINT sids [L-]label	Change sids next-lesson-pointer
QUIT	Stop the MONITOR
READ	Begin reading of student responses
REGISTER s:code	Register new student or modify registration
TRACE sids [FULL] #days	Print history of student path for #days days
UNHOLD sids	Remove a HOLD
UNNOTE sids {note# | EVERY}	Remove a NOTE
UNREGISTER s:code	Delete a student's registration
WHO sids [MISSED] LTD-label	Print s:code of students (NOT) given a unit
WHY	Request error message printout

MODE THREE

COMMAND SYNTAX	ACTION
DISPLAY {COMMANDS [1 | 2 | 3] | USERS}	Print list of commands or users
INITIALIZE {$ | $-FILE | ALL}	Empty and condition a new $-FILE
MODE {1 | 2 | ?}	Change mode or ask current mode
MTS [MTS-command]	Issue single command or enter MTS sub-mode
OFFLINE [TAIM-command]	Queue TAIM command(s) for DRIVER processing
QUIT	Stop the MONITOR
REGISTER u:code	Register a Mode One, Two or Three user
SET command# {ACTIVE | INACTIVE}	Enable or disable a command
UNREGISTER u:code	Delete a user's registration
WHY	Request error message printout

MODE TWO

COMMAND SYNTAX	ACTION
CATALOG {L|T|D}-FILE|ALL} [XREF]	Print labels of units, XREFed to LOGICs
DROP LTD-label	Remove unreferenced unit from files
EDIT ws#	Invoke system editor on a workspace
:CHANGE [lpar] ['string1'string2']	change string1 to string2 in lpar
:DELETE [lpar]	delete lines of lpar from file
:EDIT ws#	edit a different workspace
:INSERT [line#] ['text']	begin input of line(s) into the workspace
:LINE [line#]	set current-line-pointer to line#
:±n	move current-line-pointer ±n positions
:OVERLAY [lpar] ['string']	non-blank chars in 'string' replace others
:PRINT [lpar|LEN|FN]	print lines, line length, workspace name
:RENUMBER	reorder line numbers to 1, 2, 3, ...
:REPLACE [lpar] ['text']	replace whole lines
:SCAN [lpar] ['string']	find occurrance of 'string' in lpar
:SHIFT [lpar] {LEFT | RIGHT} count	shift all lines in lpar count positions
:STOP	return to TAIM
EVALUATE ws#	Examine workspace for syntax errors
FIND LTD-label	Locate references to LTD-label
LABEL ws# {LTD-label | ?}	Label workspace or request current label
MODE {1 | 3 | ?}	Change to new mode or ask current mode
PLACE {NULL | ws# | LTD-label} [INTO] ws#	Move units between files and workspaces
SAVE ws#	Copy workspace to permanent files
SETKEY ws#	Place test key in workspace ws#
+CLEAR	clear key to zeroes
+PRINT	print key
+RESET	restore previous key
+SET	begin input of response weights
+STOP	return to TAIM
+WEIGHT	set Between-Test Weight
QUIT	Terminate TAIM
WHY	Request error message printout

APPENDIX B

EXAMPLES

Included in this appendix is an example of online use of the MONITOR and a student's lesson as produced by the DRIVER.

In the MONITOR session, the user is a Mode Three user and therefore has access to all modes. He enters Mode Three automatically as he signs-on, and proceeds to display a list of users. He then moves to Mode Two and creates and SAVEs a simple Display. Transferring to Mode One, he alters a student's registration, and then sends that student a NOTE; he then sends a NOTE to all students in class "B" and signs-off. In all examples in this thesis, text in upper case represents that printed by the computer, text in lower case is that entered by the user.

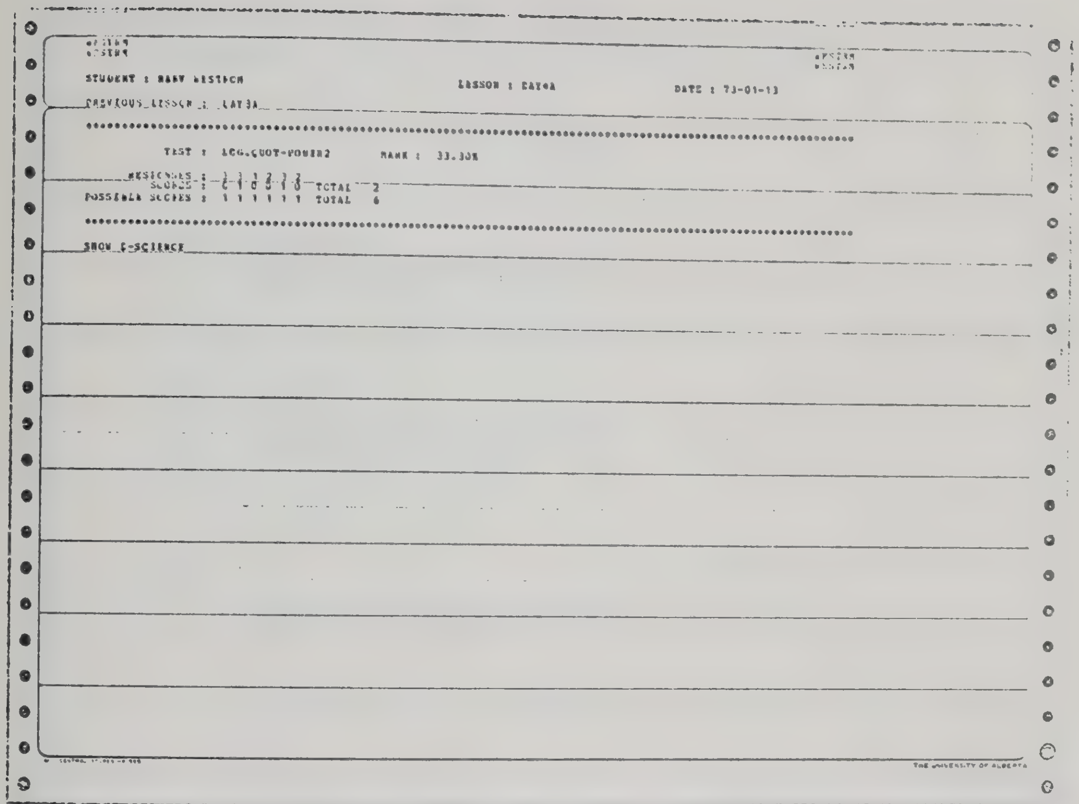
The sample of output from the DRIVER has been reduced by about 40%. The pages were originally 11" by 16". Each day of testing, each student received a printout similar to the example, but determined by his unique circumstances.

APPENDIX B

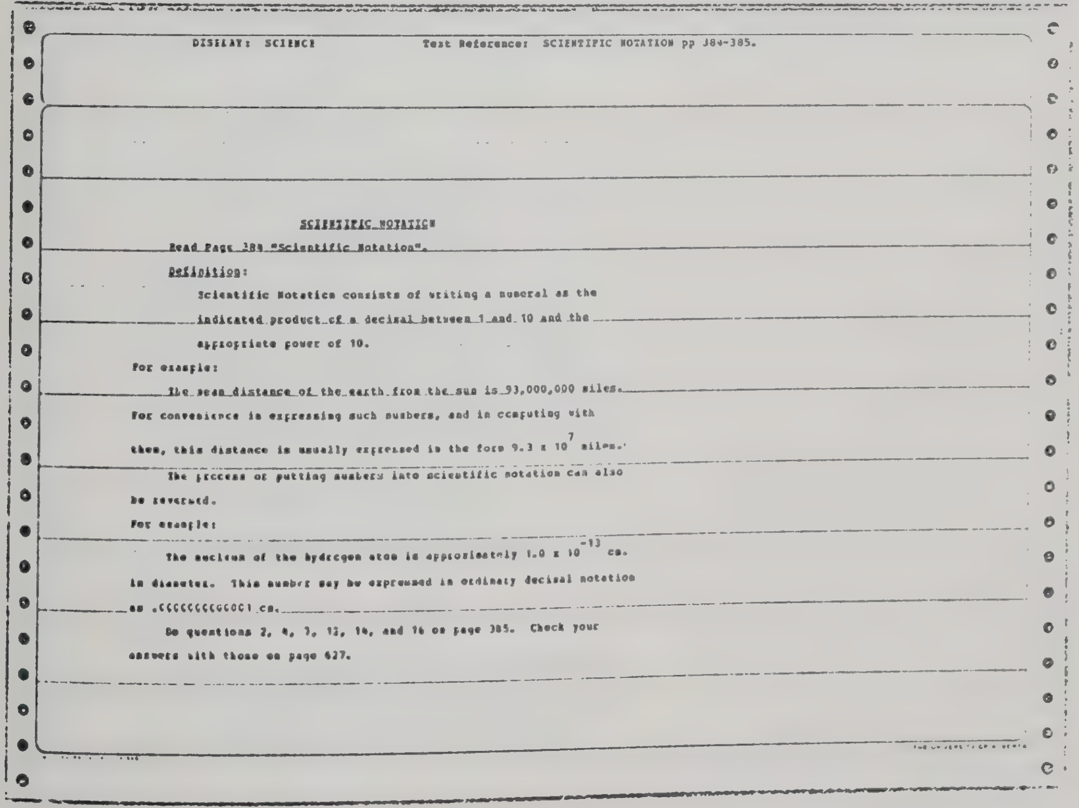
```
$run taim
#11:09.57

TAIM SYSTEM ... VERSION 1

ENTER TAIM IDENTIFICATION
zzzzzz

TAIM COMMENCED AT 11:10.14 ON 01-09-73

MODE 3 ACTIVATED


READY
disp users
***  USER   STATUS LAST-USE USES NAME
    FISHME   1     01-08-73    4 MEL FISCHER
    KIERET   2                 0 TOM KIEREN
    MOTIBO   1     01-09-73    5 ROBERT MOTIUK
    WESTRM   3     01-09-73   47 MARV WESTROM
DONE  ...  $.06


READY
mode 2
DONE  ...  $.01


READY
edit ws1
:insert 1
?this is text for the display
? <null line terminates insertion mode>
:stop
DONE  ...  $.03


READY
save ws1
NOT DONE ... 30


READY
why
WORKSPACE NOT LABELLED
DONE  ...  $.01


READY
label ws1 d-example
DONE  ...  $.01
```

```
READY
save ws1
DONE  ...  $.04


READY
mode 1
DONE  ...  $.01


READY
reg zarsky
DON  ZARSKY  IS  ALREADY  REGISTERED.  DO YOU WISH TO MODIFY
REGISTRATION? (YES OR NO).
yes
NAME    :DON ZARSKY
NAME    :ok
AGE     :12
AGE     :ok
CLASS   :A
CLASS   :b
COMMENT: GOOD BOY
COMMENT: <null line terminates registration modification>
DONE  ...  $.06


READY
note zarsky
ENTER NOTE : welcome to class b don.
DONE  ...  $.02


READY
note class-b
ENTER NOTE : don't forget ... tomorrow.
DONE  ...  $.06


READY
quit

TAIM TERMINATED AT 11:15.38 ON 01-09-73
#11:15.41 5.82 RC=0
#
```

APPENDIX B



```
STUDENT : MARY WESTECH                    LESSON : EAY4A          DATE : 73-01-13
PREVIOUS LESSON : EAY3A
****************************************************************************************
         TEST : LOG.QUOT-POWER2       MARK : 33.33%
     RESPONSES :  1 1 1 2 1 2
        SCORES :  0 1 0 0 1 0   TOTAL   2
POSSIBLE SCORES :  1 1 1 1 1 1   TOTAL   6
****************************************************************************************
SHOW E-SCIENCE
```



```
          DISPLAY:  SCIENCE              Text Reference:  SCIENTIFIC NOTATION pp 384-385.
```

SCIENTIFIC NOTATION

Read Page 384 "Scientific Notation".

Definition:

Scientific Notation consists of writing a numeral as the indicated product of a decimal between 1 and 10 and the appropriate power of 10.

For example:

The mean distance of the earth from the sun is 93,000,000 miles. For convenience in expressing such numbers, and in computing with them, this distance is usually expressed in the form 9.3×10^{7} miles.

The process of putting numbers into scientific notation can also be reversed.

For example:

The nucleus of the hydrogen atom is approximately 1.0×10^{-13} cm. in diameter. This number may be expressed in ordinary decimal notation as .0000000000001 cm.

Do questions 2, 4, 7, 12, 14, and 16 on page 385. Check your answers with those on page 627.

APPENDIX B

Do question 19.

Do the following 3 questions:

(1) The escape velocity of a rocket from the earth is 25,000 mph. Express this escape velocity in scientific notation.

(2) The time required for one addition on a certain computer is 4.4×10^{-4} seconds. Write 4.4×10^{-8} in decimal form.

(3) The radius of a hydrogen atom is 1.74×10^{-10} feet. Express 1.74×10^{-10} in decimal form.

THE UNIVERSITY OF ALBERTA

The answers are:

(1) 2.5×10^{4}

(2) .00044

(3) .000000000174

Text Reference: CHARACTERISTIC AND MANTISSA pp 385-386.

CHARACTERISTIC AND MANTISSA

Read "Characteristic and Mantissa on pages 385 - 386.

Determining the Characteristic:

To find the characteristic of a number, first think of the decimal point as being after the first significant figure of the number. The number then would lie between 1 and 10. Then the characteristic will equal numerically the number of places from this imaginary decimal point to the real decimal point, and will be positive if you count from the imaginary decimal point to the right, and negative if you count to the left.

Example 1:

Find the characteristic of the logarithm of 473.8.

Solution:

If the decimal point were after the 4, the characteristic would be 0. Since it is two places farther to the right, the characteristic is 2.

Example 2:

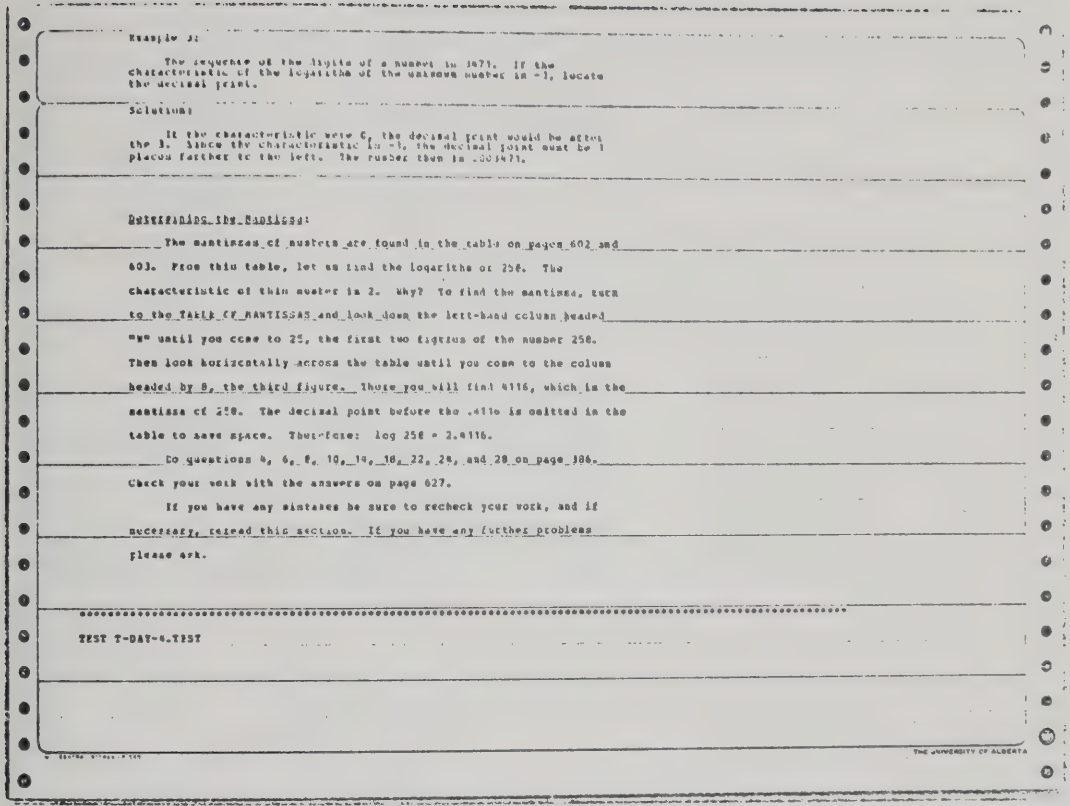
Find the characteristic of the logarithm of .0004782.

Solution:

If the decimal point were after the 4, the characteristic would be 0. Since it is 4 places farther to the left, the characteristic is -4.

If the characteristic of the logarithm of a number is given, this same rule reversed can be used to locate the decimal point in the number.

APPENDIX B



Example 3:

The sequence of the digits of a number is 3471. If the characteristic of the logarithm of the unknown number is -3, locate the decimal point.

Solution:

If the characteristic were 0, the decimal point would be after the 3. Since the characteristic is -3, the decimal point must be 3 places farther to the left. The number then is .003471.

Determining the Mantissa:

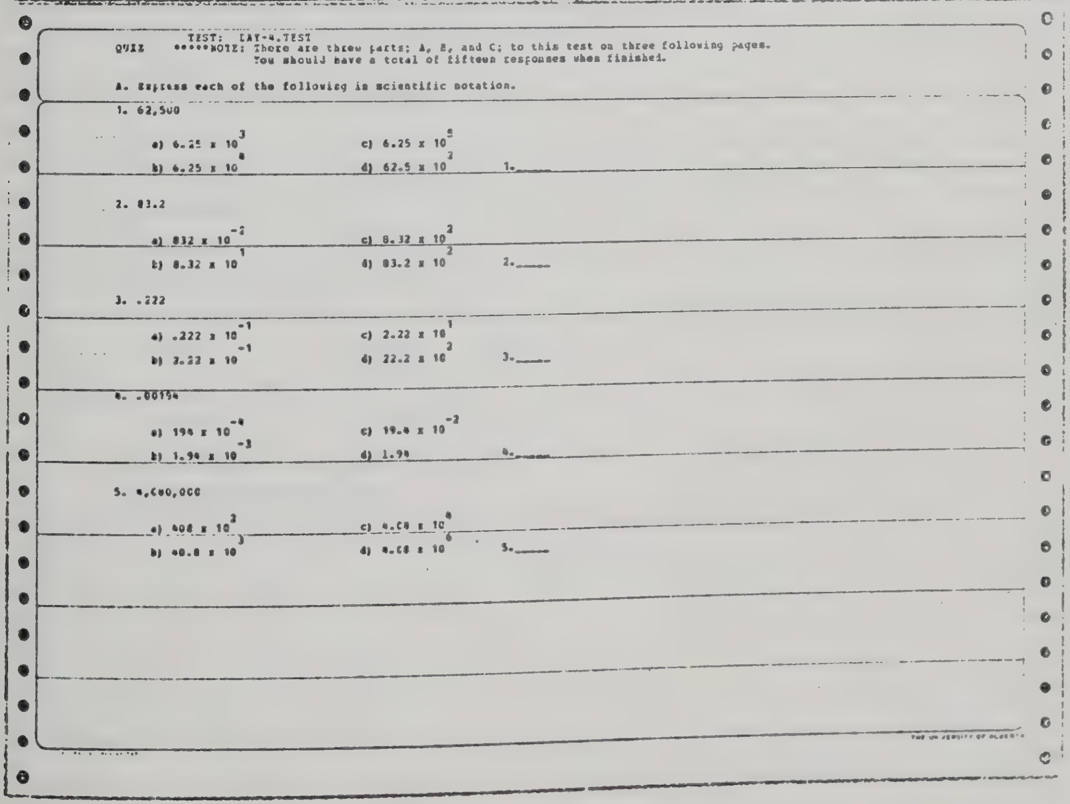
The mantissas of numbers are found in the table on pages 602 and 603. From this table, let us find the logarithm of 258. The characteristic of this number is 2. Why? To find the mantissa, turn to the TABLE OF MANTISSAS and look down the left-hand column headed "N" until you come to 25, the first two figures of the number 258. Then look horizontally across the table until you come to the column headed by 8, the third figure. There you will find 4116, which is the mantissa of 258. The decimal point before the .4116 is omitted in the table to save space. Therefore: log 258 = 2.4116.

Do questions 4, 6, 8, 10, 14, 18, 22, 24, and 28 on page 386. Check your work with the answers on page 627.

If you have any mistakes be sure to recheck your work, and if necessary, reread this section. If you have any further problems please ask.

TEST T-DAY-4.TEST

THE UNIVERSITY OF ALBERTA



TEST: DAY-4.TEST

QUIZ *****NOTE: There are three parts; A, B, and C; to this test on three following pages. You should have a total of fifteen responses when finished.

A. Express each of the following in scientific notation.

1. 62,500

   a) 6.25×10^{3} c) 6.25×10^{5}

   b) 6.25×10^{4} d) 62.5×10^{3} 1.____

2. 83.2

   a) 832×10^{-2} c) 8.32×10^{2}

   b) 8.32×10^{1} d) 83.2×10^{2} 2.____

3. .222

   a) $.222 \times 10^{-1}$ c) 2.22×10^{1}

   b) 2.22×10^{-1} d) 22.2×10^{2} 3.____

4. .00194

   a) 194×10^{-4} c) 19.4×10^{-2}

   b) 1.94×10^{-3} d) 1.94 4.____

5. 4,080,000

   a) 408×10^{2} c) 4.08×10^{4}

   b) 40.8×10^{3} d) 4.08×10^{6} 5.____

THE UNIVERSITY OF ALBERTA

APPENDIX B

B. Express each of the following in ordinary decimal notation.

6. 1.73×10^{-2}

a) .0173 c) 1.73
b) .173 d) 173 6.____

7. 4.08×10^{6}

a) .00000408 c) 408,000
b) .0000408 d) 4,080,000 7.____

8. 1.51×10^{-4}

a) .000151 c) 1,510
b) .00151 d) 15,100 8.____

9. 7.63×10^{0}

a) .763 c) 76.3
b) 7.63 d) 763 9.____

10. 2.48×10^{-3}

a) .00248 c) 2.48
b) .0248 d) 2480 10.____

C. Determine the characteristic and the mantissa for the following:

11. 8,420,000

a) 4; .9253 c) 6; .9253
b) 5; .9253 d) 7; .9253 11.____

12. .324

a) -2; .5105 c) 0; .5105
b) -1; .5105 d) 1; .5105 12.____

13. 2,810,000,000

a) 7; .4487 c) 9; .4487
b) 8; .4487 d) 10; .4487 13.____

D. Express in ordinary decimal notation

14. 3 + .9175

a) .08270 c) 82.70
b) 8.270 d) 8270 14.____

15. -5 + .6830

a) .00000482 c) 48,200
b) .0000482 d) 48,200,000 15.____

Transfer your answers to the response card. Be sure your

answers are between the numbers 1 and 4.

••

APPENDIX B

APPENDIX C

TESTS, QUESTIONNAIRES, FLOWCHARTS

This appendix contains materials that were used in the experimental procedures, and includes seven items.

ITEM 1: Three Flowcharts. First is the one designed by the writer and the Math 20 teacher and used by the Ed. C.I. 466 class in development of the instructional materials, second is the one produced by the class to depict the materials that they had constructed, third is the same flowchart after alterations for the Math 33 class.

ITEM 2: The Background Questionnaire with Responses.

ITEM 3: SUGGESTED FORMAT FOR CRITIQUE. Given to the 466 class to aid them in structuring their comments about TAIM.

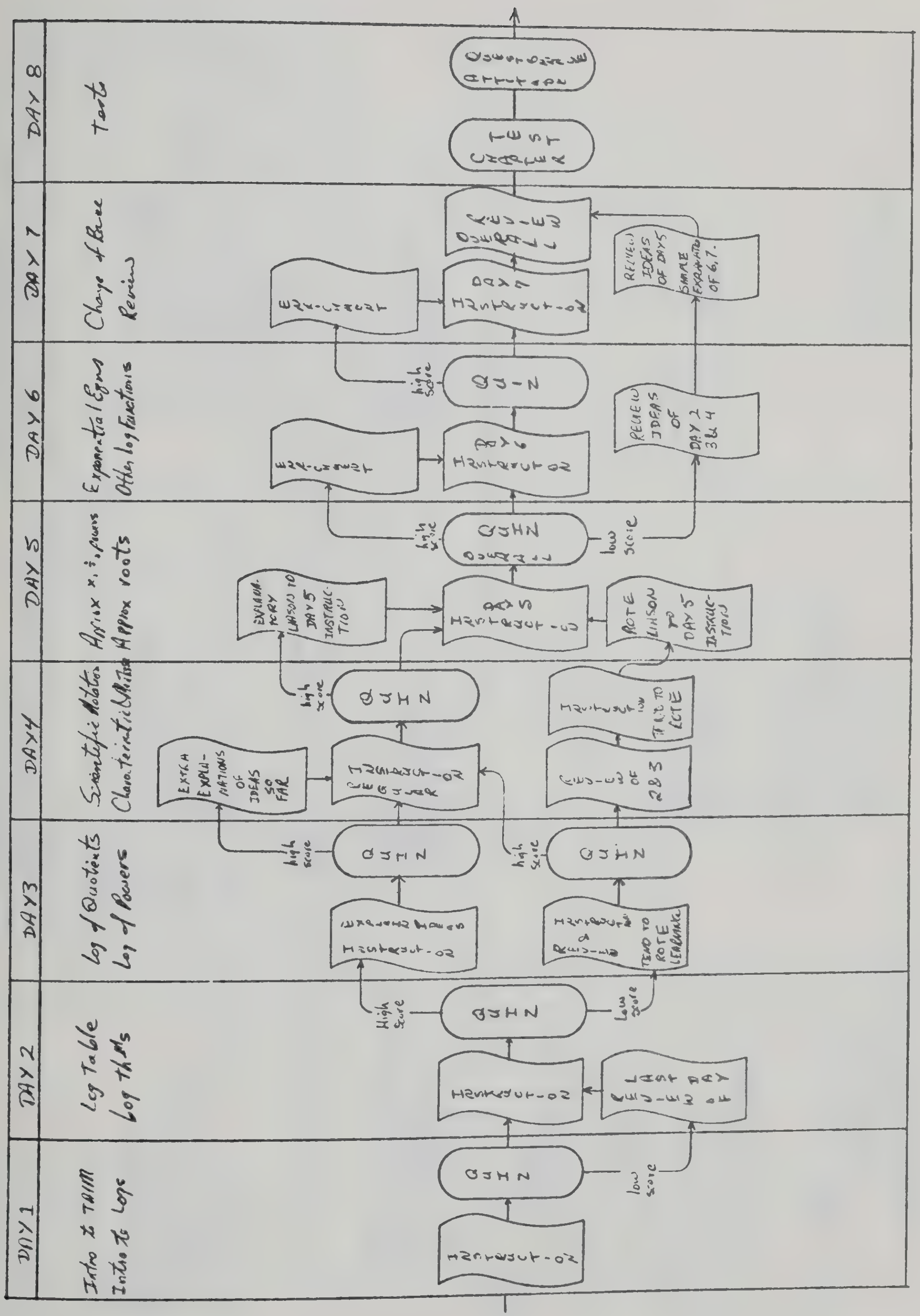
ITEM 4: Ed. C.I. 466 FINAL EXAM (TAIM Principles Exam).

ITEM 5: FOREWORD and PRE-TEST. The first Display students received and the pre-test that was given.

ITEM 6: FINAL EXAM. The Final Exam on Logarithms taken by the Math 20s and Math 33s. Items marked "*" are those selected for comparison to the pre-test.

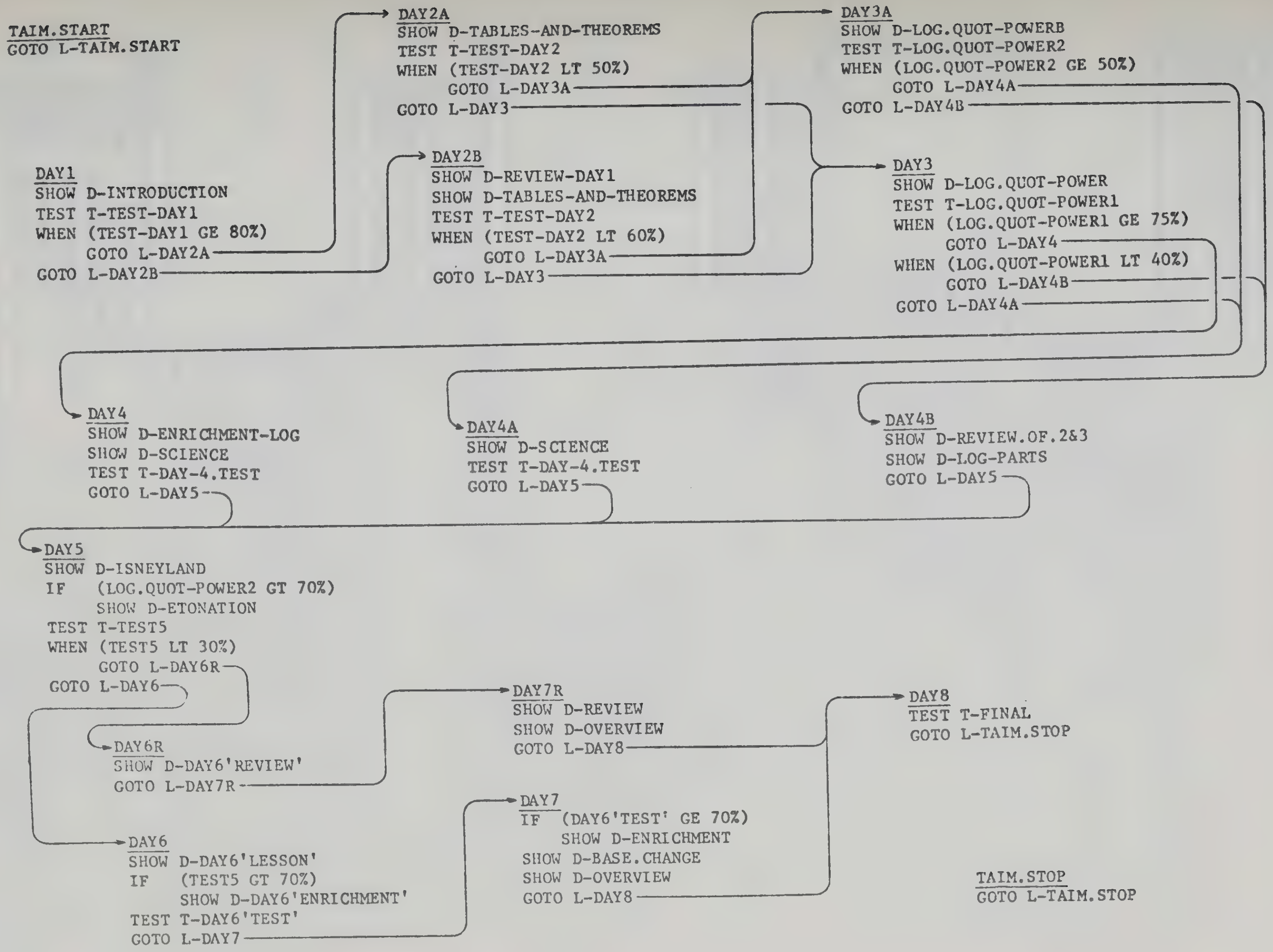
ITEM 7: Sample RESPONSE CARD.

APPENDIX C

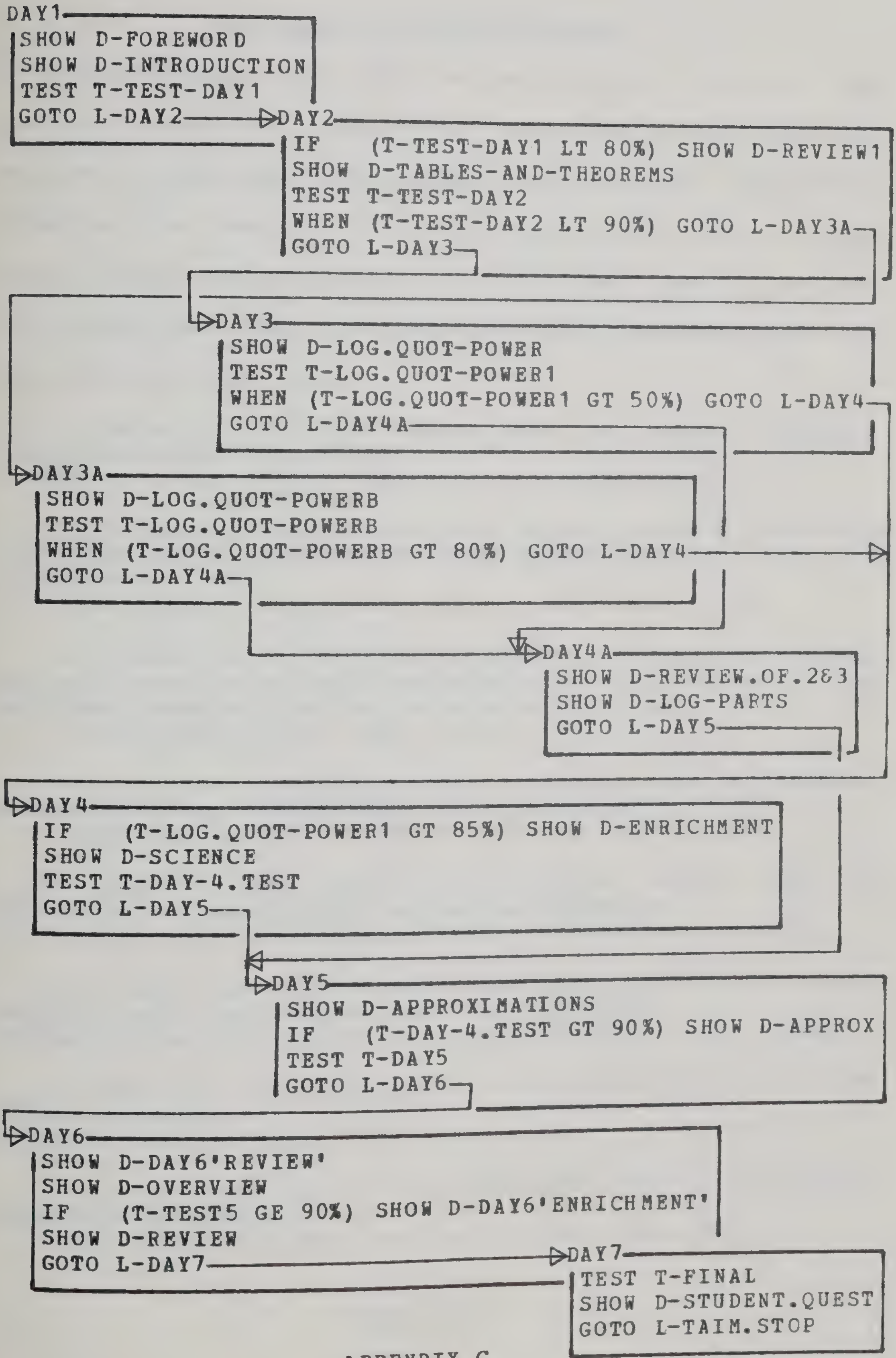

DAY 1
Intro to TAIM
Intro to Logs
INSTRUCTION
QUIZ
low score
DAY 2
Log Table
Log thms
INSTRUCTION
REVIEW OF LAST DAY
QUIZ
High score
low score
DAY3
Log of Quotients
Log of Powers
INSTRUCTION EXPLAIN IDEAS
REVIEW & INSTRUCTION
TEND TO ROTE LEARNING
QUIZ
high score
QUIZ
high score
DAY4
Scientific Notation
Characteristic Mantissa
EXTRA EXPLANATIONS OF IDEAS SO FAR
REGULAR INSTRUCTION
QUIZ
high score
REVIEW OF 2&3
INSTRUCTION TEND TO ROTE
DAY 5
Approx ×, ÷, powers
Approx roots
EXPLANATORY LESSON TO DAY 5 INSTRUCTION
INSTRUCTION DAY 5
ROTE LESSON TO DAY 5 INSTRUCTION
OVERALL QUIZ
high score
low score
DAY 6
Exponential Eqns
Other log Functions
ENRICHMENT
INSTRUCTION DAY 6
REVIEW IDEAS OF DAY 2 3 & 4
QUIZ
high score
DAY 7
Change of Base
Review
ENRICHMENT
INSTRUCTION DAY 7
OVERALL REVIEW
REVIEW IDEAS OF DAYS 5 SIMPLE EXPLANATION OF 6, 7.
DAY 8
Tests
CHAPTER TEST
ATTITUDE QUESTIONNAIRE

APPENDIX C

APPENDIX C



```
TAIM.START
GOTO L-TAIM.START

DAY1
SHOW D-INTRODUCTION
TEST T-TEST-DAY1
WHEN (TEST-DAY1 GE 80%)
     GOTO L-DAY2A
GOTO L-DAY2B

DAY2A
SHOW D-TABLES-AND-THEOREMS
TEST T-TEST-DAY2
WHEN (TEST-DAY2 LT 50%)
     GOTO L-DAY3A
GOTO L-DAY3

DAY2B
SHOW D-REVIEW-DAY1
SHOW D-TABLES-AND-THEOREMS
TEST T-TEST-DAY2
WHEN (TEST-DAY2 LT 60%)
     GOTO L-DAY3A
GOTO L-DAY3

DAY3A
SHOW D-LOG.QUOT-POWERB
TEST T-LOG.QUOT-POWER2
WHEN (LOG.QUOT-POWER2 GE 50%)
     GOTO L-DAY4A
GOTO L-DAY4B

DAY3
SHOW D-LOG.QUOT-POWER
TEST T-LOG.QUOT-POWER1
WHEN (LOG.QUOT-POWER1 GE 75%)
     GOTO L-DAY4
WHEN (LOG.QUOT-POWER1 LT 40%)
     GOTO L-DAY4B
GOTO L-DAY4A

DAY4
SHOW D-ENRICHMENT-LOG
SHOW D-SCIENCE
TEST T-DAY-4.TEST
GOTO L-DAY5

DAY4A
SHOW D-SCIENCE
TEST T-DAY-4.TEST
GOTO L-DAY5

DAY4B
SHOW D-REVIEW.OF.2&3
SHOW D-LOG-PARTS
GOTO L-DAY5

DAY5
SHOW D-ISNEYLAND
IF   (LOG.QUOT-POWER2 GT 70%)
     SHOW D-ETONATION
TEST T-TEST5
WHEN (TEST5 LT 30%)
     GOTO L-DAY6R
GOTO L-DAY6

DAY6R
SHOW D-DAY6'REVIEW'
GOTO L-DAY7R

DAY7R
SHOW D-REVIEW
SHOW D-OVERVIEW
GOTO L-DAY8

DAY6
SHOW D-DAY6'LESSON'
IF   (TEST5 GT 70%)
     SHOW D-DAY6'ENRICHMENT'
TEST T-DAY6'TEST'
GOTO L-DAY7

DAY7
IF  (DAY6'TEST' GE 70%)
    SHOW D-ENRICHMENT
SHOW D-BASE.CHANGE
SHOW D-OVERVIEW
GOTO L-DAY8

DAY8
TEST T-FINAL
GOTO L-TAIM.STOP

TAIM.STOP
GOTO L-TAIM.STOP
```



```
DAY1
SHOW D-FOREWORD
SHOW D-INTRODUCTION
TEST T-TEST-DAY1
GOTO L-DAY2

DAY2
IF   (T-TEST-DAY1 LT 80%) SHOW D-REVIEW1
SHOW D-TABLES-AND-THEOREMS
TEST T-TEST-DAY2
WHEN (T-TEST-DAY2 LT 90%) GOTO L-DAY3A
GOTO L-DAY3

DAY3
SHOW D-LOG.QUOT-POWER
TEST T-LOG.QUOT-POWER1
WHEN (T-LOG.QUOT-POWER1 GT 50%) GOTO L-DAY4
GOTO L-DAY4A

DAY3A
SHOW D-LOG.QUOT-POWERB
TEST T-LOG.QUOT-POWERB
WHEN (T-LOG.QUOT-POWERB GT 80%) GOTO L-DAY4
GOTO L-DAY4A

DAY4A
SHOW D-REVIEW.OF.2&3
SHOW D-LOG-PARTS
GOTO L-DAY5

DAY4
IF   (T-LOG.QUOT-POWER1 GT 85%) SHOW D-ENRICHMENT
SHOW D-SCIENCE
TEST T-DAY-4.TEST
GOTO L-DAY5

DAY5
SHOW D-APPROXIMATIONS
IF   (T-DAY-4.TEST GT 90%) SHOW D-APPROX
TEST T-DAY5
GOTO L-DAY6

DAY6
SHOW D-DAY6'REVIEW'
SHOW D-OVERVIEW
IF   (T-TEST5 GE 90%) SHOW D-DAY6'ENRICHMENT'
SHOW D-REVIEW
GOTO L-DAY7

DAY7
TEST T-FINAL
SHOW D-STUDENT.QUEST
GOTO L-TAIM.STOP
```

APPENDIX C

THE BACKGROUND QUESTIONNAIRE

The Background and Attitude Questionnaire given to the teachers contained thirty-five questions. The first ten requested background information, the remaining twenty-five attempted to sample attitudes. The Attitude portion of the questionnaire was administered pre and post, and is reported in Chapter Five. The questions and results obtained from the Background portion follow.

1. Have you had any experience in using computers outside of this University?
Pa-no; Pb-no; Pc-no; Pd-no; Pe-2; Pf-no; Ta-1; Tb-1.

2. Counting courses you are currently taking, how many courses have you taken which directly involved the use of computers?
Pa-1; Pb-2; Pc-2; Pd-1; Pe-2; Pf-10; Ta-0; Tb-0.

3. BASIC and APL are interpretive languages in that they will execute certain commands as they are entered. How many such interpretive languages have you used?
Pa-0; Pb-2; Pc-0; Pd-0; Pe-1; Pf-5; Ta-1; Tb-1.

4. If you have ever used a compiler type of language such as FORTRAN, COBOL, PL/1, etc.; how many?
Pa-2; Pb-0; Pc-0; Pd-0; Pe-1; Pf-5; Ta-0; Tb-0.

5. Have you ever used an on-line file editor such as MTS *EDIT, or COURSEWRITER authoring language?
Pa-no; Pb-yes; Pc-no; Pd-no; Pe-no; Pf-yes; Ta-no; Tb-no.

6. Have you previously received instruction from a computer (such as from the IBM /1500 system)?
Pa-no; Pb-yes; Pc-no; Pd-no; Pe-no; Pf-yes; Ta-no; Tb-no.

7. Have you been employed as a teacher? If yes give the number of years.
Pa-no; Pb-7; Pc-no; Pd-no; Pe-no; Pf-1; Ta-6; Tb-3.

8. How many years of University have you taken, including this year?
Pa-4; Pb-8; Pc-4; Pd-4; Pe-6; Pf-4; Ta-4; Tb-4.

Questions 9 and 10 of the background part of the background and Attitude Questionnaire were included with the post-Attitude test.

9. Report the number of the statement which best describes your attitude to discipline in the classroom.

1) Students should be encouraged to "do their own thing".
2) It is most important for students to enjoy their schooling.
3) A break once in awhile doesn't hurt anybody.
4) Students' excess energy can be put to productive use.
5) A disturbance by one child wastes the time of the whole class.

PRE: Pa-2; Pb-2; Pc-4; Pd-4; Pe-3; Pf-4; Ta-4; Tb-3.
POST: Pa-2; Pb-2; Pc-4; Pd-4; Pe-2; Pf-4; Ta-4; Tb-3.

10. Which instructional setting (below) would be most likely to accomplish the goals of a course you were teaching?

1) Lecture.
2) Frequent discussions.
3) Structured laboratory activities.
4) Planned discovery situations.
5) Complete freedom for students to learn what they wished.

PRE: Pa-4; Pb-3; Pc-4; Pd-4; Pe-4; Pf-4; Ta-4; Tb-3.
POST: Pa-4; Pb-2; Pc-2; Pd-4; Pe-4; Pf-4; Ta-2; Tb-1.

APPENDIX C

SUGGESTED FORMAT FOR CRITIQUE

A. TAIM MODE TWO

1. Individual Commands Assessment.
   - for each command, assess its usefulness, flexibility, and appropriateness; comments.
2. General Mode Two Commands Assessment.
   - are more needed?, which ones?, why?
   - general comments on commands.
3. Workspaces and Files.
   - are workspaces flexible enough?, appropriate?
   - are files flexible enough?, appropriate?
4. Learning Mode Two.
   A school principal thinking of using TAIM would want to know the "cost" of teaching Mode Two to his Junior/Senior High School staff of 25 teachers. Outline the inservice program you would use; include estimates of instruction time, lab time, and required materials.
5. General Comments.
   Assess TAIM Mode Two in terms of its ability to facilitate the construction of learning sequences for students.

B. INSTRUCTIONAL MATERIALS CONSTRUCTION

1. The Flowchart.
   Discuss the advantages and disadvantages of using an overall flowchart in the process of constructing instructional materials.
2. Unit Responsibility.
   - Should Display, Test, and Logic unit construction be the responsibility of the classroom teacher? Why or why not?
   - Should unit modification and updating be the responsibility of individual teachers? Why or why not?
3. Printer Restrictions.
   The offline printer severly limits the format and characters available for printing instructional materials. How serious are these restrictions? What can be done to overcome them?
4. Materials Production.
   Assuming you knew Mode Two thoroughly, how would you proceed to construct (from scratch) the eight days of materials produced by this class? How long would it take <u>you</u>.
5. General Comments.
   - Include any general comments you may have on the construction of instructional materials using TAIM.
   - Assess the overall feasibility of the TAIM System for individualization of instruction in Junior/Senior High Schools.

<u>APPENDIX C</u>

Ed C.I. 466 (part 2) FINAL EXAM

Answer ALL of the following questions in the answer book provided. There are 20 questions, each worth five marks for a total of 100.

1. What is the finction of the XREF parameter on the CATALOG command?
- what is the purpose of this parameter?

2. If a user entered the TAIM Mode Two command:

```
drop d-intro
```

what are the two conditions under which the command would be NOT DONE?

3. Examine the following EDITOR (or *EDIT) session:

```
:del /file
:in 1 :this is the first line:
            1      THIS IS THE FIRST LINE
:%k=lc
%O.K.
:setv=_set v=off
:in 1 'THIS is the first line'
:insert
?this is the SECOND line
?this is the FIRST line
?change 'FIRST'LAST'
?
:change@a /file 'FIRST'LAST'
:ren
:print /file
  ...
```

- what will be printed?

4. A unit labelled XYZ is in the TAIM Display file. It contains, at line three the text 'here', but should contain the text 'there'. Assuming you are "in" TAIM, list (in order), the commands you would issue to effect this change.

5. Display XYZ is in the TAIM Display file, and is to be re-named as Display ABC. D-XYZ may or may not be referenced by one or more Logic units. How would you make this change?

6. The Logic file contains three units linked with DAY3 pointing to DAY4, and DAY4 pointing to DAY5. You wish to add DAY4A with DAY3 pointing to DAY4A and DAY4A pointing to DAY5. Describe how you would insert this new Logic unit.

APPENDIX C

7. Give the syntax and describe the action of the FIND command in TAIM.

8. Give the syntax and describe the action of the EVALUATE command in TAIM.

9. In SETKEY mode, what is the action of the RESET command? How is it related to the CLEAR command?

10. Concerning TEST KEYS, define the following terms:
    - Question Weight
    - Response Weight
    - Within-test Total
    - Question Number
    - Student Response

11. The terminals we have been using do not have a key to print the bullet symbol ("•"). Some people have been using this to represent multiplication in Displays and Tests. Describe how you would cause this symbol to appear in a Display.

12. What is the action of each of the following carriage control characters:
    a) '+' plus
    b) '-' minus
    c) ' ' blank
    d) '0' zero
    e) '1' one
    f) '2' two
    g) '4' four
    - explain also the difference between relative and absolute carriage control.

13. Assuming you were "in" TAIM and that a Display called LOG.INTRO was in the Display File; explain why each of the following commands would not be executed:
    a) cat dfile xref
    b) place log.intro ws3
    c) label ws3 logs.part2
    d) print /file
    e) save d-logs.part2

14. What are the six possible statements allowed in the Logic units? - give a rough syntax of each.

15. Give the syntax and explain the semantics of the "condition" allowed in two of the Logic statements.

16. What <u>major</u> steps must the computer execute when reading in a student's response card?

17. What <u>major</u> steps are involved in assembling the student's lesson?

18. Explain the NORMAL function of the following special characters at the terminal:

a) backspace	d) cent sign
b) underscore	e) ATTN
c) exclamation mark	

19. Because of cost restrictions, we have had to construct Displays and Tests in files other than WS1, WS2, and WS3; and under IDs other than QZZ5. Explain the procedure involved in transfering a Display from a file outside QZZ5 to the TAIM Display file.

20. Concerning the command descriptions given in this course; explain the difference between syntax and semantics. Include an explanation of the use of brackets [...], braces {...}, upper and lower case, and underlining; and the distinction between keywords and parameters.

APPENDIX C

FOREWORD

For the next eight school days, you will be receiving a computer printout such as this one. These printouts are designed to lead you through the concepts you should know about logarithms. Today, everyone has the same printout; but as each of you get high or low marks on various quizzes and tests - you will receive different "lessons".

All of the things you need to know about logarithms are NOT present in these printouts. However, if you read the printouts and your text carefully, you will encounter all of the necessary ideas.

At one time or another, almost everyone is bound to run into something he or she does not understand. If this happens to you, please use the following procedure:

1) Re-read the printout and your text to see if you can resolve the problem yourself. If you cannot, ...

2) If you have a friend working on the same problem, and if he or she isn't too busy, ask for help.

3) Failing all else; ask the teacher.

PRE-TEST

Before we begin the TAIM instruction, please answer the following PRE-TEST on the computer card supplied.

1. Using the tables on pages 602-603 of your text, calculate log(7.52)
   a) 0.8762 b) 56.5 c) 565 d) 8762
   e) none of these

2. Which of the following is equivalent to $\log(3x^2)$?
   a) $3\log(x^2)$ b) $\log(3) + \log(x) + \log(2)$
   c) $6\log(x)$ d) $\log(3) + 2\log(x)$

3. What is the value of log to the base three of 27?
   a) 1/9 b) 3 c) 9 d) 81

4. Express 325.66 in scientific notation.
   a) 32566/100 b) 325 + 1/3 c) 3.2566 x 10
   d) 300 + 20 + 5 + 6/10 + 6/100

5. What is the characteristic of log(123.4)?
   a) 1 b) 2 c) 3 d) 4 e) none of these

When you have finished these questions, please hand your computer card to the teacher. Thank you for taking this introductory test; you may now proceed with today's lesson.

APPENDIX C

LOGARITHMS FINAL EXAM

Each of the questions that appear below are followed by numbers. Circle the number in frcnt of the correct answer. There is ONLY one correct answer for each question. When you have completed the test, transfer your answers onto the white card that is provided.

*

1. 0.00000235 expressed in scientific notation is:

   1) 235×10^{8}. 2) 2.35×10^{-8} 3) 235×10^{6} 4) 2.35×10^{-6}

2. Assuming that Log(2) = 0.30103
   use this fact to determine log(8).

   1) 0 + 0.90309 2) 1 + 0.20412 3) 0 + 0.60206 4) 0 + 0.90304

3. If a=bc, then what is log(b) equal to?

   1) log(c) - log(a) 2) log(a) - log(c) 3) log(a)/log(c) 4) log(a) + log(c)

4. $\mathrm{Log}_3(27^{1/3}) =$

   1) 1. 2) 2. 3) 3. 4) 4.

5. The solution set for

   $3^{x+2y} = 3$ and $2^{x+2y} = 2^{y}$ is:

   1) (-3,1) 2) (1,0) 3) (-1,1) 4) (-1,1/2)

6. If i) $\log_X 27=3$, and ii) $\log_Y 8=3$, and iii) $\log_Z 16=4$, which of the following is true?

   1) $X=Y=Z$ 2) $X=Y$ 3) $Y=Z$ 4) $Y>Z>X$

7. What is the value of $10^{\log_{10} X}$?

   1) $\log(10^X)$ 2) 10^X 3) 10 4) $\log_X(10)$ 5) X

8. What is the log of the square root of $(10^{-2+0.341})$?

   1) $-11 + 0.705$ 2) $-1 + 0.1705$ 3) $-1 + 0.4682$ 4) 0.4682 5) -0.4682

9. If $\log_{10}(6) = 0.778$ and $\log_{10}(4) = 0.602$, what approximately is $\log_4(6)$?

   1) 0.129 2) 1.29 3) 1.48 4) 1.73

*
10. In the expression "log(x) = 2.4371" the .4371 is called the:

   1) base 2) characteristic 3) mantissa 4) $\log_{10}$

11. $\text{Log}(Y^R \bullet X^N / Z^S)$ may be expressed as:

1) $Y\log R + Y\log N - Z\log S$ 2) $Y\log R \bullet X\log N - Z\log S$

3) $R\log Y \bullet N\log X / S\log Z$ 4) $R\log Y + N\log X - S\log Z$

12. Log 1000 =

1) 1.000 2) 2.000 3) 3.000 4) 4.000

13. The statement $5^4 = 625$, written in logarithmic form, would be:

1) $\log_4 625 = 5$ 2) $\log_5 4 = 625$ 3) $\log_4 5 = 625$ 4) $\log_5 625 = 4$

14. $\text{Log}(10^{0.4562}) =$

1) 0.4562 2) $10^{0.4562}$ 3) 10 4) 1.4562 5) 4.562

15. If $\log_b 10 = p$, and $\log_b 2 = q$, then $\log_b 20 =$

1) pq 2) $p + q$ 3) p^q 4) q^p 5) $2(p+q)$

*
16. What is the value of the log of the cube root of B?

1) 3•log(B) 2) (log(B))/3 3) cube root of (log(B))

4) (log(B))/(log(3)) 5) log(B/3)

17. If $\log_m 7 = 1/3$, then the value of m is:

1) 49 2) 2.333 3) 21 4) 343 5) 1.913

18. Log(3X • 3X) =

1) $\log_3 X^2$ 2) 2log3X 3) $\log 9^{2x}$ 4) 2log9X

19. The value of $5^{\log_5 125}$ is:

1) 3 2) 5 3) 15 4) 125

*
20. $\text{Log}_2 2^3$ =

1) 2 2) 3 3) 3log2 4) 9

21. If log(21.7) = 1.3365, and log(2.16) = 0.3345, then log(21.7/2.16) =

1) 1.6710 2) 0.0020 3) 1.0020 4) 3.9955

22. If $\log_{10} X = 2$, then X =

1) 2 2) 4 3) 12 4) 20 5) 100

23. For the value $\log_b X$, which of the following is true?

1) X may be any real number 2) b may be any real number

3) b must be >0, but should not be 1 4) b may be <0 when X<0

5) b must be 1, 2, 3, or 10

24. $((4.83/3.96)^5)$ =

1) 0.0454 2) 2.699 3) 3.433 4) 0.4310

25. If $\log_X Y = 1$, then X =

1) 0 2) 1 3) 10 4) Y

26. What is the domain of {(1,3),(2,3),(3,3)}?

1) {3} 2) {3,3,3} 3) {1,2,3} 4) {8,9}

27. Log ((3.11 • 2.42)/(6.60)) =

1) 0.0571 2) 0.2038 3) 0.9285 4) 1.0522

28. 2.14 • 10^{-4} =

1) 21,400 2) 0.0214 3) 0.000214 4) 21.4000

29. Which of the following is true?

1) log(7^2) = 7 • log(2)

2) log($(3•4)^2$) = 2•log(3•4)

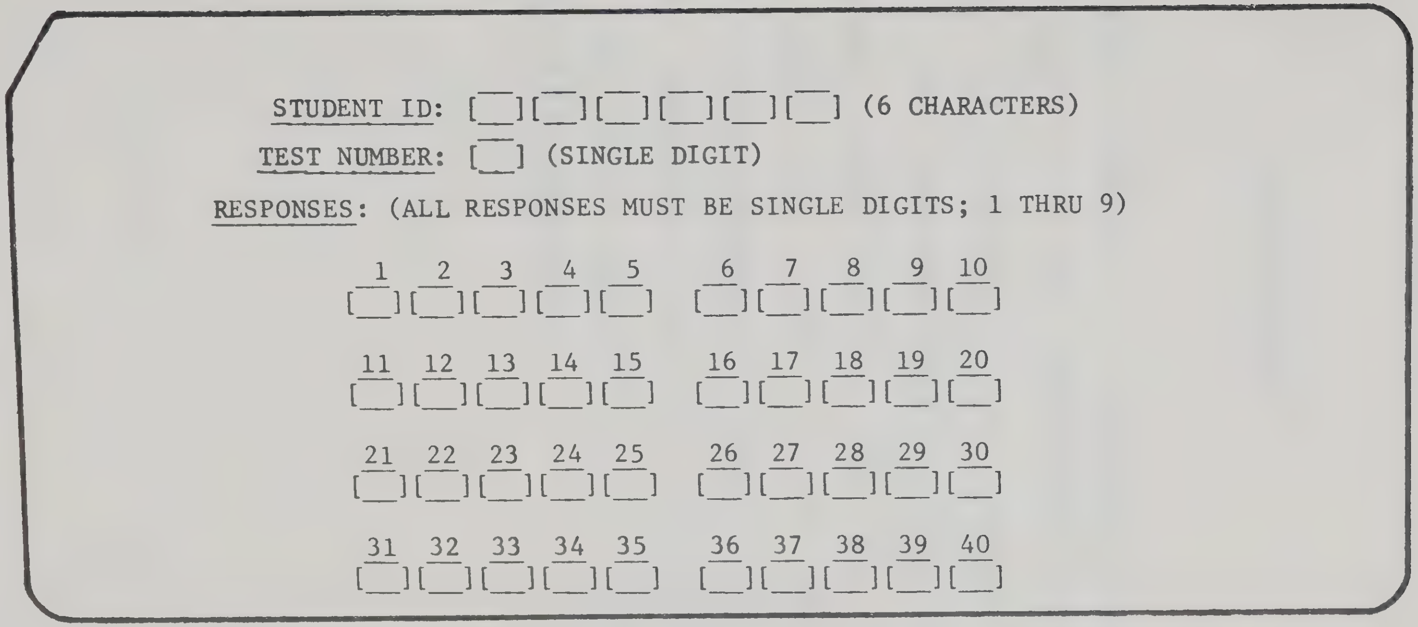
3) log((sq. root of 2)/2) = 1/2 • log(2) -2

4) log(3 • sq.root of 2) = 1/2 • log(3•2)

*
30. The mantissa of 2,810,000,000 is:

1) 9 2) 10 3) .4487 4) .04487

APPENDIX C



STUDENT ID: [ ][ ][ ][ ][ ][ ] (6 CHARACTERS)

TEST NUMBER: [ ] (SINGLE DIGIT)

RESPONSES: (ALL RESPONSES MUST BE SINGLE DIGITS; 1 THRU 9)

1	2	3	4	5	6	7	8	9	10
[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
11	12	13	14	15	16	17	18	19	20
[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
21	22	23	24	25	26	27	28	29	30
[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]
31	32	33	34	35	36	37	38	39	40
[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]	[ ]

APPENDIX D

THE MTS EDITOR

This Appendix consists of a reformatted version of the printout that the Ed. C.I. 466 participants received at the second meeting. It is included for two reasons: to provide an example of the printout and to provide an abbreviated description of the MTS EDITOR. The Mode Two command EDIT invokes this same file editor so that the descriptions actually apply to the implemented version of TAIM as well. The reader is cautioned that the descriptions are not complete as they were intended to be supplemented by discussions and other printouts.

EDITOR INTRODUCTION

Within the MTS system is a program called the EDITOR. It has a variety of uses with the arrangement of characters within a file or workspace. A workspace can be considered as an extendable piece of paper upon which any characters can be placed. It is arranged (as are most arrangements of characters) into lines. Each line has an associated number (for example, this is line 7 of this paragraph) and the numbers are always kept in ascending order. Unlike a printed page, lines can always be inserted between existing lines of a file, or added to the beginning or end of the file. The EDITOR can be used on any number of files, but only with one file at a time.

With many of the EDITOR commands, it is necessary to specify the line or lines on which the commands are to be executed. This (if present) is either the first, or the first and second parameters after the keyword of the command, and is called the "LINE PARAMETER(S)" or "lpar" for short. Whenever the abbreviation "lpar" is used in a syntactic description of a command, it should be read as:

lpar is {line# | line#1 line#2 | /FILE} where

line# is an acceptable line number:

1) a value between -99999.999 and 99999.999 (inclusive) with at most three decimal places,

2) *F indicating the first (smallest numbered) line in the file,

3) * indicating the "current line" in the file, or

4) *L indicating the last (largest numbered) line in the file.

line#1 line#2 are each line#s (as above) separated by a blank, and arranged so that line#1 < line#2.

/FILE indicates all lines in the file, and is equivalent to "*F *L".

If line# is used, this indicates that the command is to be applied only to the specified line. If line#1 line#2 is used, the command is to be applied to all lines numbered between and including these two line#s. If /FILE is used, the command is to be applied to all lines in the file..

When editing a file, there is always a "current line". When editing starts, this is line 1 of the file; however the current line (*) is changed by some of the editor commands. In all of the editor commands, if "lpar" is specified as optional, and is omitted, the "current line" is assumed. Similarly, if line# is optional and is omitted, the current line is assumed by default.

EDITOR COMMAND MODIFIERS

Some of the editor commands may be modified by appending the keyword with a "COMMAND MODIFIER". This modifier must by connected to the keyword or to a previous modifier with NO intervening blanks. Three acceptable command modifiers are

@NV	- at no verification
@A	- at all
@X	- at hexadecimal

APPENDIX D

Many editor commands "verify" the results they have produced by printing out the line or lines they have changed or located in the file. This printout can be suppressed by using the @NV modifier on the command.

Example: change@nv ... This modifier is not applicaple to all commands; for example, print@nv does not make sense. The @A and @X command modifiers will be explained in context with the editor CHANGE command.

EDITOR COMMANDS

While not all editor commands are explained in this section, those given should be sufficient for the majority of editing work you will be required to do. A complete description of the MTS EDITOR is contained in the manual "MTS Volume 1: MTS and Computing Services" pages 289 - 332; available at the University Bookstore.

Putting material into a file can be easily accomplished with the INSERT command.

:INSERT [line#] ['string']

The parameter "line#" directs where the insertion(s) are to be made. If line# is the number of a line already in the file, insertion is made AFTER the line numbered line#. If no line with number line# is currently in the file, insertion begins AT line line#. The square brackets around line# indicate that it is optional. Recall then that if it is omitted, the "current line" or the value "*" is assumed.

The parameter 'string' is the text of the line to be inserted, with a DELIMITER on either side. The delimiters are not part of the text and are not inserted. The delimiter may not be any character contained in the string, nor may it be any of the characters: blank, digit, *, or /. Any other characters (eg: + & ¬ ' " : , .) may be used. If 'string' is omitted, fast insertion mode is entered. That is, the user is prompted with a ? to enter successive lines of text until he enters a line with no characters (a NULL line); in this mode, the delimiters are not used. If 'string' is present, the inserted line is verified (printed) unless the @NV modifier was used on the INSERT keyword. The last line inserted always becomes the "current line".

```
EXAMPLES:      (1) :insert 5 'this line'
               (2) :in@nv :I don't know:
               (3) :IN *L
               (4) :i
```

Getting lines into a file may seem difficult, but getting them out is very easy.

:<u>D</u>ELETE [lpar]

As usual, if lpar is omitted, the current line is assumed.

```
EXAMPLES:      (1) :delete /file
               (2) :de *f
               (3) :DE 3 5
               (4) :d
```

Any line may be replaced using the REPLACE command:

:<u>R</u>EPLACE [lpar] ['string']

If lpar is omitted, * is assumed; if 'string' is omitted, the line(s) to be replaced are printed and the user prompted

with a ? to enter a replacement for each. If both lpar and 'string' are given, all lines in lpar are replaced by 'string'. Whenever 'string' is given, the replaced line is verified unless the @NV modifier was used.

```
EXAMPLES:      (1) :replace 3 "this is new line 3."
               (2) :re@nv 4 5   ...
               (3) :r
```

It is often desirable to PRINT certain portions of the file with:

```
     :PRINT [lpar | LEN | FN]
```

The command to PRINT lpar behaves as expected. PRINT LEN will print the length of the current line, and PRINT FN will print the name of the file being edited.

```
EXAMPLES:      (1) :print /FILE
               (2) :PR *L
               (3) :pr 5.5 10
               (4) :p *f
```

After editing a file, the numbers of the lines may become unwieldy. These numbers can be rearranged with the the command:

```
     :RENUMBER
```

This command causes the lines to be renumbered from one with an increment of one (ie: 1, 2, 3, 4, ...). Because of the way this command has been implemented, the ATTENTION key is NOT disabled: therefore, DO NOT PRESS ATTN while this command is executing. This is the ONLY command requiring this restriction.

As was previously mentioned, when editing of a file first begins, the current line is set at line 1 (or the

first line) of the file. The current line is directly controlled by two commands: LINE and ±n. It is indirectly changed by the INSERT, EDIT, and SCAN commands (as well as others not mentioned here).

:LINE [line#]

causes line line# to become the current line, and further causes that line to be printed unless the @NV modifier is used. With this command, the current line may be absolutely positioned.

```
EXAMPLES:      (1) :line 5
               (2) :l@nv *l
```

:±n

allows relative positioning of the current line. The command :+5 causes the fifth next line to become the current line (and be printed). The command :-3 causes the third previous line to become the current line (and be printed). The + is optional. Suppose a file had 9 lines numbered 1 to 10 with line 5 missing. Then:

```
:line 4     would make line 4 the current line, then
:2          would make line 7 the current line, then
:-1@nv      would make line 6 the current line
```

(suppressing verification).

When editing a file, it is sometimes desirable to stop editing that file and start editing another file. This may be accomplished with the command:

:EDIT filename

Where "filename" is the name of the new file to be edited.

APPENDIX D

The current file is released and the new file FILENAME brought in. The first line with a number ≥ 1 becomes the current line which is printed (unless @NV was used on the EDIT command).

```
EXAMPLES:      (1) :edit -tfile
               (2) :ed@nv ws3
```

When you wish to stop editing, simply issue the command:

:STOP

Besides the REPLACE command, there are two other commands to change the contents of a line already in the file. These are OVERLAY, and CHANGE.

The OVERLAY command allows the user to everlay a line with replacement characters:

:OVERLAY [lpar] ['string']

If "lpar" is omitted, "*" is assumed; if "'string'" is omitted, the line(s) are typed out and the user prompted with a ? to enter an overlaying string for each. If both lpar and string are present, all lines in lpar are overlayed with 'string'. The @NV modifier may be used to prevent verification in all cases.

Any character in the original line overlayed with a blank is left unchanged. Characters overlayed with any other character are changed to the new character.

```
EXAMPLES:  (1)  user :overlay 2       (2)  :o@nv /FILE '0'
            computer :ABCDEFG         (3)  :over 3 8
                user : x y z          (4)  :ov ' x y z'
            computer :    2  AxCyEzG  (5)  :o
```

APPENDIX D

The CHANGE command has two possible strings:

:CHANGE [lpar] ['string1'string2']

Where prime (') is the delimiter as discussed with 'string' previously; string1 is a string of characters as they currently appear in a line, and string2 is a replacement for string1. lpar defaults to *. If 'string1'string2' is omitted, this parameter from the last issued CHANGE command is assumed. The change command works only for the first occurrance of string1 found in lpar unless the modifier @A (at ALL) is used. @NV may be used to suppress verification of the results of the change.

```
EXAMPLE: Suppose line 1 of a file contained "ababa". Then:
      :ch 1 'b'bx'       would produce      "abxaba"   then
      :ch@a 1 :ab:y:     would yield        "yxya"     then
      :ch 1 'yx''        would give         "ya"       .
```

Note in the last command, that a set of characters was changed to NO characters.

The change command can also be used to obtain some of the special characters available on the offline printer but not available on the terminal. This involves the use of the @X modifier. This procedure will be explained in the next printout. Some of these characters are:

± ≤ ≥ ≠ 0 1 2 3 4 5 6 7 8 9 + − ( ) { } [ ] • ▪ □ .

To locate a portion of text, one can use:

:SCAN [lpar] ['string']

As usual, lpar defaults to *; if 'string' is omitted, the 'string' from the last issued SCAN command is assumed. Only the first occurrance of 'string' is found unless the @A

modifier is used. If the string is found, the last line in which the string was found becomes the current line. If 'string' is not found in lpar, "STRING NOT FOUND" is printed and the value of * is unchanged.

```
EXAMPLES:      (1) :sc 1 10 :don't:
               (2) :scan@a /file
               (3) :s
```

Lines can be shifted left or right with the command

:SHIFT [lpar] {LEFT | RIGHT} count

Where LEFT or RIGHT indicate the direction of the shift, and count is a positive integer indicating the distance of the shift. Note that either LEFT or RIGHT must be specified, as must count. The @NV modifier may be used to suppress printing of the shifted lines. Lines shifted right are padded with blanks on the left. Lines shifted left lose any characters shifted past position 1 of the line.

```
EXAMPLES:      (1) :shift 3 r 5
               (2) :sh@nv /file r 1
```

There are also a number of keywords that can be set in the editor.

:SET LNR=OFF
will cause lines to be printed without their linenumbers in front.

:SET V=OFF
will have the same effect as using the @NV modifier on every subsequent command.

:SET CC=ON
will cause lines to be printed using carriage control (except +) when LNR=OFF.

:SET LNR=ON :SET V=ON :SET V=OFF
would reverse the actions of the above command settings.

APPENDIX D

EDITOR COMMAND SUMMARY

COMMAND PROTOTYPE

```
CHANGE [lpar] ['string1'string2']

DELETE [lpar]

EDIT filename

INSERT [line#] ['text']

LINE [line#]

±n

OVERLAY [lpar] ['string']

PRINT [lpar | LEN | FN]

REPLACE [lpar] ['text']

SCAN [lpar] ['string']

STOP

RENUMBER

SHIFT [lpar] {LEFT | RIGHT} count
```

DAY2 LAB

Now that you have some prepared curriculum (prepared as a result of last day's lab), and have learned some EDITOR commands, you are ready to place your text into a file.

1) Signon to your regular MTS ID and create a file using the MTS CREATE command. For example $CREATE curric If the computer will not let you create a file because of "INSUFFICIENT DISC SPACE", you may have to DESTROY some of your *BASIC files.

2) Start the MTS EDITOR with the command $RUN *EDIT The program will ask you for the name of the file you wish to edit. Enter the name of the file you created ("curric" in the above example).

3) Normally, the computer converts all input characters to upper case. To have it accept lower case, the device command "%k=lc" must be issued. The computer should respond with "%OK."

4) Issue the INSERT command and start inputting lines.

APPENDIX D

Use other EDITOR commands as necessary to place your text into your file. Don't worry about spacing the lines on the page (this will be easier done using carriage control explained next day); but do keep the length of your lines under 90 characters.

5) After inputting and correcting all the lines you wish, issue the EDITOR RENUMBER and STOP commands. After you have returned to MTS (# prefix) issue the %RESET device command to reinstate upper case conversion.

6) $RUN a batch job to print out your file. Use the PRINT=TN and the 'DELIVER TO EDUC 8' options.

Your terminal printout should look something like this:

```
a
#MTS (LA33-0009)
#sig ccid
#PASSWORD?
?******
#ON AT  <date>  ON  <time>  LAST ON AT  <time>  ON <date>
#cr curric
#FILE "CURRIC" HAS BEEN CREATED.
#run *edit
:ENTER FILE NAME:
:curric
:%k=lc
%OK.
:in
?  <your insertions>
   <other EDITOR commands>
:ren
:stop
# <stop time>
#%reset
%OK.
#run *batch
 REMOTE  BATCH:  PLEASE ENTER JOB
>$sig ccid print=tn 'deliver to educ 8'
>passwd
>$list curric
>$signoff
>¢
END OF FILE
 ASSIGNED   JOB   #  <job number>
 JOB <job number>  (PRIORITY  12) IS IN POSITION <position>
#  <stop time>
#sig
# <statistics>
```

<u>APPENDIX D</u>

APPENDIX E

U OF A RATE SCHEDULE

The following page is a reproduction of a schedule produced by the Department of Computing Services. It details the charges levied for all types of computer resources at the University of Alberta.

The University of Alberta, Edmonton

May 1, 1972

computing services rate schedule

The following tables show the rates for various computer resources effective May 1, 1972. To calculate the cost of a computing activity use the following formula:

Final Cost = **Computer Resource Charges** (Table 1) × **Priority Factor** (Table 2) × **Client Factor** (Table 3) + **Charges for Other Services, if any** (Table 4)

Table 1
Computer Resources

Resource	Rate	Units	Notes
CPU Time	$300.	hour	
VM-CPU[1]	$3.00	page-hour	
Magnetic tape	$3 20	hour	
Disk	$0 50	page-month	exempt from
Terminal connect	$1 50	hour	Priority Factor
Cards read	$0 50	thousand cards	
Cards punched	$4 00	thousand cards	
Lines printed	$0 40	thousand lines	
Pages printed	$4.40	thousand pages	
SOBF	$0.25	job	exempt from Priority Factor
OS/360 CPU when memory			
≤118K bytes	$375	hour	
≤200K bytes	$450	hour	
≤300K bytes	$525	hour	
≤400K bytes	$600	hour	
≤500K bytes	$675	hour	
OS/360 lines printed	$0.40	thousand lines	
OS/360 cards read	$0.50	thousand cards	
OS/360 card punched	$4 00	thousand cards	

Computer resource charges are the sum of the rates for the resources used times the number units of resources used.

Table 2
Priority Schedule

Priority Class	priority factor	Execution Priority[2]	Print Priority[2]	Remarks
LOW (L)	.7	0	0	FIFIWC[3] Held til 9 p m
NORMAL (N)	1.0	See table 2a		In priority order
HIGH (H)	1.3	13	11 (> 50 pgs)	FIFIWC[3]
			13 (≤ 50 pgs)	FIFIWC[3]
RUSH (R)	2.0	14	12 (> 50 pgs)	FIFIWC[3]
			14 (≤ 50 pgs)	FIFIWC[3]
TERMINAL	1 3	Not applicable		
	1 0		May-August and after 7 p m	

Priority factors apply to all computer resources except terminal connect, disc storage and SOBF runs.

Table 2a
Execution and Print Priority for Normal (N) Jobs

CPU Time	Pages Printed (P)				
in minutes (T)	P≤50	50<P≤100	100<P≤200	200<P≤400	400<P
T ≤ 1	12	11	10	9	8
1 < T≤3	11	10	9	8	7
3 < T≤6	10	9	8	7	6
6 < T≤10	9	8	7	6	5
10<T≤15	8	7	6	5	4
15<T	7	6	5	4	3
Print Priority	10	9	8	7	6

Table 3
Client Classification Table

Type of Client	Client factor
University projects funded from operating or research accounts	0 4
Non-profit organizations	1 25
Contracted research projects	2 0
Commercial projects	2 0

Table 4
Charges for Other Services
No client factor applies to these services

Type	Item	Rate	Units
Keypunching	Student[4]	free	
	Rush	$5 60	hour
	General	$4 65	hour
GRID	Graphics terminal	to be set	hour
Calcomp Plotter	Plotting	$10.00	hour
	Paper	$0 10	foot
Optical Scorer	Processing	$0.01	sheets/pass
	Operator	$3.25	hour
IBM 2741 Terminal	with Data Set	$140.11	month
	with Acoustic Coupler	$109.11	month
	Hardwired	$99.11	month
IBM 029 Keypunch		$73.21	month
Card storage	Locker	$2.00	month
		$10 00	year
	Rack space	free. permit	4 months
Programming consultants GSB-255	Advice (limited to half-hour maximum)	free	
Consulting		Negotiable fee	
Contract Programming	Initial cost	$25 00[5]	
	Assistant Analyst	$8 00	hour
	Analyst	$10 00	hour
	Senior Analyst	$12 00	hour
	Manager	$15 00	hour

[1]VM-CPU is the integral of virtual memory with respect to CPU time. It is a measure of the core storage reserved for your program.
[2]higher priority number jobs are performed first
[3]First In, First Initiated Within Class

[4]When related to a university course requiring computing.
[5]Designed to recover cost of proposal and estimate. Not applicable unless project proceeded with.

APPENDIX E